Telecommunication Systems (2025) 88:57
https://doi.org/10.1007/s11235-025-01290-2

Fair switch selection for large scale software defined networks in next generation internet of things

Mohammad Shahzad[1] · Lu Liu[2] · Ajay Kaushik[3] · Irum Bibi[4] · Nacer Edine Belkout[5] · Mahmood ul Hasan[6]

Accepted: 3 April 2025

Abstract

Software Defined Networking has been pivotal in enabling on-demand resource utilization and is poised to have an incredible impact on the next phase of the Internet of Things. Its ability to furnish a versatile and expandable network framework is instrumental in accommodating the overwhelming surge of IoT devices and applications. The combination of static mapping and the dynamic flow of traffic over time and space creates an uneven distribution of loads across SDN controllers. Dynamic migration is a solution aimed at rectifying this imbalance by redistributing the load between SDN controllers. Communication for control between switches and controllers becomes burdensome when the matching rules are absent from the table. Our prior research has addressed this issue by employing burst aggregation focused on consolidating similar destinations to reduce the control overhead. In this study, our focus is on ensuring fairness during migration and selecting the appropriate switch. We model a fair switch selection (FSS) algorithm tailored for large-scale software-defined networks. Unlike traditional methods using packets as a basis, FSS utilizes bursts as its input. This model prioritizes bursts considering both their distance and destination, ensuring that switches select bursts with the highest priority to maintain quality of service. Our research delves into evaluating the performance of the proposed algorithm in comparison to four baseline algorithms: round robin, exhaustive search, multi-protocol TCP (MPTCP), and random search. Through extensive simulations, we analyze experimental results based on cost, performance, packet loss, average throughput, and execution time. Experimental results demonstrated a reduction in packet loss by 30% with an average 25% throughput improvement.

Keywords Internet of Things · Migration · Scheduling · Software defined networking

1 Introduction

As traditional networks face challenges that hinder their flexibility, efficiency, and scalability, the Software Defined Networking (SDN) paradigm is rapidly emerging as an alternative solution for the Internet of Things (IoT). Managing the exponential growth of connected devices, data volumes, and network traffic poses significant challenges for traditional networks. Scaling such networks often involves laborious manual configuration and provisioning processes, which are time-consuming and prone to errors. To overcome these challenges and others, SDN has emerged as a dominant force in network architecture, significantly reducing

✉ Mohammad Shahzad
mshahzad@warwickshire.ac.uk

Lu Liu
l.liu3@exeter.ac.uk

Ajay Kaushik
a.kaushik@derby.ac.uk

Irum Bibi
fragrant_bareeze@hotmail.com

Nacer Edine Belkout
Belkout.naceredine@gmail.com

Mahmood ul Hasan
mahmood.mscs@gmail.com

[1] Warwickshire College and University Centre, Royal Leamington Spa, UK

[2] Department of Computer Science, University of Exeter, Exeter, UK

[3] University of Derby, Derby, UK

[4] Islamabad, Pakistan

[5] Department of Computer Science, University of Science and Technology Houari Boumediene, Algiers, Algeria

[6] Project National Industrial training Institute, TUV Rheinland Arabia, Khobar, Kingdom of Saudi Arabia

Published online: 27 April 2025

reliance on traditional hardware-based approaches. In addition, SDN provides enhanced visibility into network traffic by offering centralized monitoring and analytics capabilities. This centralized control is particularly valuable in IoT environments, where efficient provisioning, configuration, and monitoring of numerous devices are crucial. SDN, which is responsible for managing large networks that incorporate cloud resources and data centers [1], encounters the challenge of overloaded controllers requiring load shifting or balancing. When controllers become overloaded, it can lead to performance degradation, heightened latency, and potential disruptions. Various factors can contribute to overloaded controllers, including increased network traffic, inefficient algorithms, suboptimal resource utilization, and security attacks.

Over the past few years, researchers have extensively investigated load balancing in SDN, as well as migration techniques between controllers, driven by the aim of reducing control overhead. The potential solutions to address these challenges include scaling the controller infrastructure, employing efficient and optimized algorithms, enhancing resource allocation mechanisms, implementing traffic offloading strategies, and leveraging monitoring and analytics capabilities. The recent emergence of SDN and its significance in the context of IoT [2] has garnered significant attention from researchers and various industries, leading to widespread adoption of this innovative architecture. SDN in the realm of Next Generation IoT (NG-IoT) [3] has experienced a significant shift at a macro level, leading to a growing recognition of the benefits associated with adopting this novel network management approach. Numerous SDN frameworks have been proposed for various domains such as next-generation smart cities [4], cloud computing [5], edge computing [6], and 6G [7], among others. These frameworks have specifically focused on load balancing and migration challenges in the context of SDN-IoT. The quest for load balancing in networks introduces network delays and service disruptions. Conversely, a significant amount of controller capacity remains unused, resulting in idle resources. Even when controllers are not operating at full capacity, they still consume the same amount of energy and power as controllers running at full load. Network engineers strive to maximize the utilization of available resources, but there remains a considerable gap that needs to be addressed and filled. Achieving load balancing in a distributed multi-controller deployment presents a new challenge in the face of dynamic traffic. Load balancing [8] has become a crucial benchmark for SDN to maintain its competitive edge in the IT market. Unbalanced load distribution adversely affects the controller throughput, leading to increased response time. Switch migration [9–11] emerges as a promising solution for resource balancing. However, existing approaches often overlook migration costs and unwanted control overhead. These concerns can be addressed by implementing improved migration and scheduling schemes, which have the potential to mitigate these parameters effectively.

In the realm of SDN, managing communication encounters certain difficulties. Unlike conventional networks where individual devices autonomously make choices, SDN adopts a centralized intelligence that hinges on interactions between the control and data planes. The issue emerges when dealing with NG-IoT, as the vast number of devices push the boundaries of SDN controllers. The communication control link connecting controllers and corresponding switches is witnessing a surge in activity, often resulting in absent entries within the table. This work delves into the process of selecting switches for migration to an SDN controller, with the primary goal of enhancing load balancing while minimizing migrations and ensuring fairness in switch selection for migration. Through a comprehensive analysis of various switch selection methods and a thorough review of state-of-the-art literature, numerous critical insights have been unveiled. Several misconceptions concerning traffic prediction and scalability issues within large-scale networks have been identified. Additionally, the omission of Quality of Service (QoS) considerations during load balancing, insufficient comparisons with existing research methods, and limited scalability evaluations have been noted. In response, the proposed FSS algorithm is contributing to switch selection migration with a focus on fairness, link utilization, and QoS factors. The FSS algorithm not only reduces migrating switch numbers but also employs criteria such as destination and distance for switch selection.

Furthermore, the paper addresses the concern of control overhead reduction by introducing the concept of feeding bursts to OpenFlow switches. The FSS algorithm undergoes rigorous evaluation and comparison with four baseline algorithms, distinguishing itself from common practices that often involve comparing just two algorithms. Notably, FSS exhibits significantly lower computation times than its counterparts, effectively tackling the challenge of control overhead in SDN. The integration of bursts aids FSS in overcoming this control overhead challenge within the context of next-generation IoT. This scheme not only enriches our understanding of switch selection methods but also uncovers overlooked aspects of the field. These findings hold considerable relevance for large-scale SDN NG-IoT applications, serving as a solid foundation for further research and advancement.

While numerous researchers are dedicated to addressing this significant concern, none have ventured into the concept of consolidating packets into bursts based on their common destination, aiming to curtail control overhead. This innovative aggregation model strives to diminish control overhead by dynamically grouping packets destined for the same endpoint. The process involves implementing a burst aggregation

algorithm at the edge router, closely monitoring its queue. By tracking packets and their intended destinations, we systematically assemble them into bursts. Notably, this burst assembly procedure [12] employs specific thresholds, commencing at the assembly of 5 packets within the queue and progressively escalating up to 50 packets. Additionally, the data transmitted to the switches in this context will be in the form of bursts. Choosing the right switches in SDN for NG-IoT deployments for efficient and reliable communication is pivotal. This selection directly impacts the network performance metrics to ensure the required bandwidth and QoS for the IoT devices generating different types of data. An approach to address the fairness in the network, fairness amongst the selection of switches is applied in this paper.

For better network utilization, scheduling algorithms are proposed to cater to the bursty nature of SDN NG-IoT. Identifying migration switches helped in gathering the necessary information required, in case, there can be more than one switch in a domain to be migrated to immigration controller.

Hence, a new procedure is introduced and algorithm to help with the accurate and fair switch selection. To sustain the dynamic selection of switches while enhancing load balancing and migration capabilities, it becomes imperative to impose restrictions on migration to avoid unnecessary consumption of controller resources. The scheme presented hinges on three factors to determine switch selection during instances of contention: the total number of nodes, the nodes yet to be traversed, and the nodes successfully reached. The total nodes contribute to fairness considerations, the remaining nodes elevate burst priority as the destination nears, and the successful nodes impact link utilization. This paper introduces an innovative approach to switch selection for seamless migration, ensuring both QoS preservation and equitable network performance. Notably, the switches are supplied with bursts of data instead of individual packets.

The structure of the paper is Sect. 2 provides a comprehensive review of the existing scheduling algorithms in SDN and highlights their constraints. Section 3 delves into the proposed design and its corresponding methodology. Section 4 places emphasis on the implementation details. Lastly, Sect. 5 offers a conclusion to encapsulate the findings and insights.

2 Related works

SDN control plane assumes the responsibility of overseeing network components such as routers and switches. The control plane manages various tasks on these devices, encompassing network configuration, routing determinations, and the enforcement of network policies. Particularly in expansive networks like NG-IoT equipped with multiple controller installations, maintaining equilibrium within the control plane is of paramount importance. This equilibrium in the control plane is essential to ensure the continual availability of SDN controllers. Furthermore, by optimizing the distribution of control plane load, scalability is achieved, enabling the network to seamlessly accommodate the continuous addition and removal of dynamic devices. The meticulous design of the SDN control plane becomes imperative to effectively distribute the load among controllers. This approach not only averts bottlenecks that could impede network traffic but also guarantees a harmonized network operation.

In [13], researchers have put forth a strategy for achieving load equilibrium within SDN networks via a predictive switch migration scheduling technique. This novel approach anticipates forthcoming network traffic loads by leveraging machine learning algorithms, and subsequently orchestrates the distribution of switch migrations among controllers to uphold load balance. The prediction process considers factors such as network topology, switch capacity, and controller workloads. The predictive model, utilizing the ARIMA method, is benchmarked against alternative load balancing methodologies, demonstrating superior load equilibrium and a reduction in network congestion.

A novel on-demand scheduling algorithm [14] is proposed for intelligent routing load management within an SDN framework. The primary objective is to enhance the efficiency of network resource usage and alleviate network congestion through the dynamic adjustment of routing paths. This is accomplished by deploying a predictive model that assesses network load and strategically selects optimal routing paths for individual data flows, aligning with load balancing principles. The outcomes of this approach yield an enhanced user experience and heightened network efficiency. The industry 4.0 landscape heavily relies on the industrial internet, where network control necessitates utmost reliability and minimal latency. The interconnectedness of IoT devices through the industrial internet facilitates the supervision and control of industrial systems. In this context, ensuring exceptionally low end-to-end waiting times for data transmission stands as a significant challenge. The coexistence of multiple data streams further complicates matters. Notably, while business flows regulate a relatively small portion of data volume, other flows such as interactive and sensing business flows contribute substantial data to the network. These latter flows are processed by conventional switches, which employ a store-and-forward mechanism that leads to higher bandwidth consumption, buffer overflow, and subsequent inefficiencies.

In [15], the authors have sidestepped conventional switches by leveraging the capabilities of an SDN framework to tackle the challenges associated with diverse service flows. To accommodate the dynamic network topology, their scheduling policy capitalizes on the northbound API of the SDN controller. Edge and intermediate switches are strategically employed to transmit data at specific intervals,

mitigating the need for queuing. The selection of the optimal path with minimal latency is facilitated through an enhanced Lagrangian relaxation algorithm. The SDN controller defines path rules in the flow table, and the effectiveness of this approach is confirmed through mathematical analysis and simulations.

To address latency concerns in cloud computing, an alternative solution is presented in the form of fog computing for IoT users. Capitalizing on the SDN architecture's adeptness in managing network flows, the robust switches within it can concurrently serve as fog devices or gateways. However, these fog devices are susceptible to attacks, with TCP SYN flood attacks being prevalent. These attacks involve malicious nodes generating half-open TCP connections on fog nodes and gateways to disrupt their functioning. In [16], the authors propose employing SDN to counter TCP SYN flood attacks in IoT-fog networks. They introduce a security-aware task scheduler named FUPE, which utilizes a fuzzy-based multi-objective particle swarm optimization approach to aggregate optimal computing resources. The outcomes of their study showcase FUPE's efficacy, leading to an 11% and 17% enhancement in average response time, respectively.

Emerging applications with extensive bandwidth requirements and substantial data traffic have sparked the quest for innovative traffic scheduling solutions. [17] delves into the domain of scheduling algorithms within the context of SDN, commencing with a comprehensive introduction to the SDN architecture. Subsequently, a multi-path algorithm designed for SDN load balancing is introduced, and its effectiveness is assessed through simulations conducted on the Mininet platform. A comparative analysis with traditional equal-cost multi-path routing is performed. The paper underscores the merits inherent to the proposed algorithm, which encompasses enhanced network performance, congestion mitigation, and optimized resource utilization. Furthermore, it elucidates the intricacies of algorithm deployment within an SDN environment and expounds on its performance assessment, carried out via simulations replicating real-world scenarios.

In [18], the authors delve into the intricacies surrounding the dynamic mapping and scheduling of service function chains (SFCs) within the context of SDN and network function virtualization (NFV) networks. SDN introduces a division between control and data planes, yielding greater control, while NFV involves the utilization of general-purpose servers to execute network functions. The concept of SFC revolves around the integration of these virtualized network functions to craft specific network services. The study delves into the collaborative exploration of dynamic virtual network function (VNF) mapping and scheduling, aiming to elevate the performance of service provisioning. The primary objective is to achieve load balancing while upholding QoS standards. To address this, the authors formulate the problem of VNF mapping and scheduling as a mixed integer linear programming (MILP) approach.

The inclusion of containerized services within mobile edge clouds presents an avenue for large-scale, real-time applications to achieve swift response times and minimal latency. Simultaneously, the introduction of live container migration seeks to facilitate dynamic resource management and accommodate user mobility. However, as network topology expands and migration requests surge, the existing cloud data center algorithms geared towards planning and scheduling multiple migrations prove insufficient for the vast scope of scenarios inherent to edge computing on a large scale. As a result, [19] utilizes an SDN controller to conceptualize resource conflicts during live migrations through the creation of a dynamic resource dependency graph. They put forth an iterative algorithm grounded in Maximal Independent Set (MIS) principles, designed for the efficient planning and scheduling of numerous migrations. This algorithm makes use of actual mobility traces from taxi and telecom base station coordinates. The outcomes demonstrate the algorithm's proficiency in orchestrating multiple live container migrations within expansive edge computing environments. Given the current proliferation of data centers and their continuous deployment, conventional techniques for traffic scheduling give rise to issues such as congestion and load imbalances.

In the study presented in [20], a novel approach is introduced that combines two dynamic scheduling algorithms: the Genetic Algorithm and Ant Colony algorithms. These algorithms are strategically implemented within the framework of SDN architecture to attain a comprehensive global perspective. The primary emphasis of this algorithm is directed towards addressing "elephant flows"—substantial data streams. This is accomplished by precisely calculating the optimal path and subsequently implementing rescheduling techniques. When compared against only two baseline algorithms, the proposed approach asserts its capability to effectively diminish the maximum link utilization while simultaneously augmenting overall bandwidth.

The challenges linked to SDN control layer traffic, encompassing issues such as single-path limitations, QoS concerns, congestion, and delays, are thoughtfully tackled in [21]. This work introduces a QoS-oriented global multi-path traffic scheduling algorithm as a proposed solution. Through the application of deep reinforcement learning on traction links, an algorithm for calculating link weights is presented. Following this, a traffic scheduling algorithm that leverages these link weights is put forth, and subsequently, it is integrated with the preceding section. In today's context, the smart industry is witnessing an upsurge in demands for effective information processing and management. However, sensors within this landscape encounter challenges related to congestion and scheduling. SDN emerges as a viable solution

for industrial wireless networks and seamlessly integrates with the realm of industrial IoT. To address the imperative of enhanced scheduling and resource utilization, [22] introduces the concept of Dynamic Resource Management and Scheduling (DRMS). In this context, the algorithm "Earliest Deadline First" (EDF) is implemented to effectively manage congestion and optimize packet handling. The EDF approach facilitates a prioritized scheduling mechanism for sensors. The authors assert that this approach significantly reduces network delay times and enhances energy efficiency.

The study under consideration presents certain constraints. While traffic prediction is assumed to be precise, the unpredictability of traffic patterns poses a challenge. The potential scalability of the approach within extensive SDN networks is not thoroughly explored. Additionally, the way the proposed methods address QoS issues concerning SDN load balancing lacks in-depth elaboration. Furthermore, the showcased works offer limited comparative assessment against established algorithms or contemporary techniques. This limitation arises because the focus is on evaluating our algorithm's performance in comparison with four baseline state-of-the-art algorithms.

Table 1 offers an inclusive comparison of state-of-the-art methods in the domain of SDN switch migration, scheduling, and load balancing, stressing their key focus areas, proposed solutions, comparison metrics, strengths, and limitations, with a specific stress on the unique contributions of the proposed Fair Switch Selection (FSS) algorithm.

3 System design and implementation

3.1 Flow table management and mitigation strategies

Within the empire of SDN, a prevalent challenge arises from the absence of entries in the flow table, a pivotal component overseen by the SDN controller. This flow table is instrumental in determining the routing of incoming packets. Another complication surfaces when considering bursts traversing longer distances, as the dearth of entries in the flow table can give rise to a heightened likelihood of burst loss. This phenomenon detrimentally impacts both network throughput and the judicious utilization of network resources. Consequently, network administrators must adopt strategies to mitigate this issue, such as pre-populating the flow table with common routes or enhancing the controller's responsiveness to new flow requests. Additionally, leveraging machine learning algorithms to predict and proactively install necessary flow entries can further alleviate these challenges. Ensuring seamless communication between the SDN controller and switches is also crucial, as it can reduce the latency in flow setup. Thus, addressing these challenges is vital for maintaining the efficiency and reliability of SDN networks.

3.2 Fair switch selection methodology

To mitigate these issues, a solution is presented under the moniker of "Fair Switch Selection" (FSS). This scheme is designed to facilitate the apt selection of a switch within the SDN framework for migration purposes. The underlying objective is to instill equity within the network, ensuring a balanced allocation of resources and the equitable treatment of network flows. This is achieved by discerning the appropriate switch based on specific metrics and subsequently initiating the migration process. The FSS methodology is grounded in three pivotal factors that govern switch selection. Firstly, the load on each switch is meticulously evaluated to ensure that no single switch becomes a bottleneck, thereby promoting a more uniform distribution of network traffic. Secondly, the latency between switches and the SDN controller is considered to minimize delay in packet forwarding and enhance overall network performance. Lastly, the historical performance data of switches is analysed to predict future behavior and reliability, ensuring that the selected switch can handle the anticipated traffic load efficiently. By integrating these factors, the FSS scheme aims to optimize switch utilization, reduce the likelihood of burst loss, and improve the overall robustness of the SDN infrastructure.

3.3 Controller load calculation

The performance and fairness of switch migration in SDN depend significantly on the accurate estimation of controller load. In the FSS algorithm, we define the controller load L_C as a weighted function of three key parameters: link utilisation between the switch and controller, controller processing latency, and buffer occupancy. These parameters collectively reflect a controller's instant working status and stress level.

The controllers' load is determined by the formula (1):

$$L_C = \alpha \cdot U_{link} + \beta \cdot D_{proc} + \gamma \cdot B_{occ} \tag{1}$$

where

U_{link}: Current link utilisation between a candidate switch and the controller (normalised between 0 and 1),

D_{proc}: Measured processing latency or response time of the controller in handling requests,

B_{occ}: Real-time buffer occupancy level at the controller,

α, β, γ: Weighting coefficients that determine the relative impact of each factor (default values: 0.4, 0.4, and 0.2 respectively).

These metrics are collected in real-time from periodic feedback from the controllers and switches. The metrics are normalised for comparison and updated dynamically to

Table 1 Key contributions of state-of-the-art research with FSS

Paper	Focus	Solution proposed	Comparison metrics	Strengths	Weakness
Filali et al. [13]	Predictive switch migration	ARIMA-based predictive scheduling for load balancing	Load balance and network congestion	Reduce congestion and effective load prediction	Unpredictability and scalability of traffic
Ma et al. [14]	On-demand scheduling for SDN	Intelligent routing load adjustment	Efficiency, congestion and resource usage	Enhanced resource usage and adaptive routing	Limited scalability evaluation
Song et al. [15]	Industrial internet hybrid flows	Lagrangian relaxation algorithm for flow scheduling	throughput and latency	Effective service flows and optimized latency	Restricted to certain industrial IoT use cases
Javanmardi et al. [16]	Security-aware task scheduling	FUPE: Fuzzy-based task scheduling for IoT-fog	Response time, security, and resource usage	Enhanced security and reduced response times	High computation overhead
Lu [17]	Multi-path traffic scheduling	Multi-path algorithm for load balancing	Congestion and resource utilization	Reduced congestion and optimize the resources	Lacks real-world validation
Li et al. [18]	Service function chains (SFC)	MILP for VNF mapping and scheduling	Load balance and Quality of Service (QoS)	Better service provisioning	Computationally exhaustive
He et al. [19]	Large-scale container migration	Maximal Independent Set for scheduling	Migration cost, response time	Efficient large-scale migration planning	Restricted to edge computing situations
Li et al. [20]	Elephant flow scheduling	Genetic and Ant Colony algorithms for scheduling	Bandwidth and link utilization	Lower link utilization, optimized bandwidth	Attentions only on high-volume flows
Proposed FSS (This Work)	Fair switch selection, QoS	Burst aggregation, fairness in switch migration	Throughput, packet loss, execution time, cost	Lower overhead, QoS preservation, scalability	Further QoS prioritization needed

follow the changing network state. This multi-dimensional approach ensures that switch migration is not determined by controller request rates alone, but also by underlying infrastructure bottlenecks and stress.

During the selection, FSS chooses the controller with the lowest L_C value that meets resource availability constraints. By incorporating this model, the FSS algorithm enhances decision accuracy, avoids overloading of single controllers, and ensures long-term performance and fairness in large-scale SDN IoT networks.

3.4 Computational complexity analysis

To determine the feasibility and scalability of the new proposed FSS algorithm, we now show the analysis of its computational complexity.

Let:

S be the total number of switches in the network,

C be the number of available SDN controllers.

In the FSS algorithm, for each target switch, the algorithm examines all reachable controllers to determine which migration target is optimum in consideration of the computed controller load L_C. Therefore, in the worst case, the total number of comparisons is:

$$O(S * C) \tag{2}$$

This implies that the algorithm has linear complexity with the number of controllers and switches. Because the number of controllers employed in a typical SDN deployment is relatively small (ordinarily on the order of 3–10), linear complexity with the number of switches ensures that the algorithm is highly efficient in practice.

Moreover, the individual components of controller load—link utilization (U_{link}), processing delay (D_{proc}), and buffer occupancy (B_{occ})—are available from controller

telemetry data with minimal overhead in real time. The calculation of burst priority, which is lightweight arithmetic based on hops traveled and hops remaining, has minimal computational expense.

As compared to exhaustive search algorithms with exponential or factorial time complexity, FSS offers a more scalable solution without compromising performance as shown by the evaluation results. This makes FSS an applicable and scalable solution for large-scale SDN IoT applications.

3.5 Definition of migration cost

Migration cost is a vital building block in the switch migration decision-making process within SDNs. An erroneous migration decision may result in performance reduction, increased packet loss, or wasted controller resources. To provide an effective measure of the cost of migrating a switch S_i to a controller C_j, the migration cost $M_{i,j}$ is defined in (3) as an aggregate that has three vital building blocks:

$$M_{i,j} = \lambda \cdot P_{loss} + \mu \cdot D_{reconfig} + \nu \cdot B_{overhead} \tag{3}$$

where

P_{loss}: Estimated packet loss during migration,

$D_{reconfig}$: Reconfiguration delay in re-establishing the control channel and updating flow tables,

$B_{overhead}$: Bandwidth overhead required to transfer state information and update the control path,

λ, μ, ν: Tunable weighting parameters representing the relative importance of each component (default values: 0.5, 0.3, and 0.2 respectively).

Packet Loss (P_{loss}): This is a measure of the packets that might have been discarded during the switchover period. It is a function of the time to tear down and reestablish control sessions and is of primary concern in real-time or mission-critical applications.

Reconfiguration Delay ($D_{reconfig}$): This accounts for the delay introduced while installing flow entries and synchronization between the switch and the new controller. It is typically a function of network latency and controller responsiveness.

Bandwidth Overhead ($B_{overhead}$): This quantifies the additional traffic introduced by the migration process itself, including state transfers, control messages, and protocol-level handshakes.

These parameters are normalized to equal units and for comparison. The total migration cost $M_{i,j}$ is then utilized by the FSS algorithm to select the switch-controller pair that not only improves fairness and load balancing but also minimizes performance loss during migration.

Incorporating this cost function improves the robustness of FSS decision-making and ensures that switch migrations are efficient and effective under dynamic traffic conditions.

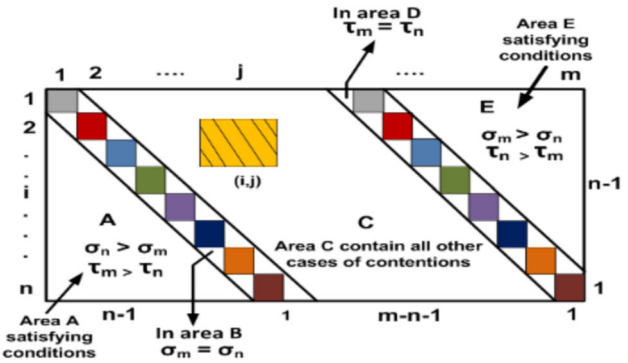


Fig. 1 FSS selection rules for bursts G_m and G_n

3.6 Prioritization and scheduling of network bursts

Consequently, this approach not only addresses the immediate challenges but also contributes to the long-term sustainability and scalability of the network.

H_i = Total hops.

σ_i = successful hops.

τ_i = remaining hops.

H_i corresponds to the notion of fairness within the network, ensuring that resources are allocated in an equitable manner across different network flows. This parameter is crucial for maintaining a balanced distribution of traffic, preventing any single flow from monopolizing network resources. τ_i augments the burst's priority as it approaches its destination, effectively increasing the likelihood that packets will reach their endpoint without unnecessary delays or loss. This prioritization becomes increasingly important in scenarios where timely delivery is critical, such as real-time communications or data streaming services. Lastly, σ_i is linked to the utilization of network links, providing an indication of how heavily a particular link is being used. By monitoring this metric, the system can make informed decisions to avoid overloading any single link, thereby optimizing overall network performance.

The selection process for switches is exemplified in Fig. 1. In this illustration, two bursts, G_m and G_n, reach the switch, with burst m holding a higher priority than burst n. The figure demonstrates how the system evaluates each burst based on the parameters. The matrix in the figure is divided into distinct segments, each representing the total number of hops a burst undertakes from source to destination. This segmentation helps in visualizing the path each burst will take through the network, highlighting potential points of congestion or delay.

For instance, when bursts G_m and G_n arrive at a switch, the system must decide which burst to prioritize for forwarding. Given that burst G_m has a higher priority (indicated by τ_i), it is more likely to be forwarded first, ensuring that it reaches its destination promptly. Burst G_n, having a lower priority,

might be delayed slightly, but the system ensures that this delay does not significantly impact overall network fairness (H_i). The matrix segments also allow for a detailed analysis of how each burst's path impacts network utilization (σ_i), guiding the selection of switches that can handle the traffic without becoming bottlenecks.

This comprehensive approach ensures that the network operates efficiently, balancing the competing demands of fairness, priority, and utilization. By carefully considering H_i, τ_i, and σ_i, the FSS scheme can dynamically adjust to changing network conditions, providing a robust solution to the challenges of burst management in SDN environments. Consequently, the FSS methodology not only enhances immediate network performance but also contributes to the long-term stability and scalability of the network infrastructure, ensuring that it can adapt to future demands and technological advancements.

The network is partitioned based on the values of σ_i (the number of successful nodes) and τ_i (the number of remaining nodes). These regions, designated as A, B, C, D, and E, cover all conceivable scenarios that might arise. Each region is carefully analyzed to determine the most effective scheduling strategy for network bursts, ensuring optimal performance and resource utilization.

Region A: Successful Nodes vs. Remaining Nodes.

Region A is characterized by a comparison between bursts n and m regarding the number of successful nodes traversed. If burst n has traversed a greater number of successful nodes than burst m, it adheres to rule 1 for scheduling. Rule 1 prioritizes bursts with a higher count of successfully traversed hops, as they have utilized more extensive resources. This prioritization ensures that resources already invested in these bursts are not wasted. Conversely, if burst m has more remaining nodes to traverse compared to burst n, it indicates that burst n is closer to its destination. In such cases, burst n will be scheduled based on rule 2, which grants priority to bursts in closer proximity to their destination servers.

Region B: Equal Successful Nodes.

In region B, the scenarios involve bursts m and n having an equal number of successful nodes. When this occurs, the priority decision is influenced by additional factors. Burst m may supersede burst n due to a higher priority level, even though both bursts have traversed an equal number of successful nodes. The decision is made based on predefined priority levels assigned to the bursts, ensuring that higher-priority traffic receives the necessary attention and resources.

Region C: Complex Scenarios.

Region C encompasses more complex scenarios that might involve a combination of successful nodes and remaining nodes in different proportions. These scenarios require a nuanced approach to scheduling, considering both the progress made by the bursts and their remaining journey. The decision-making process in this region is guided by a combination of rules 1 and 2, along with additional criteria that might be relevant to specific network conditions or policies.

Region D: Equal Remaining Nodes.

In region D, the focus shifts to bursts m and n having an equal number of remaining nodes. When the remaining nodes are the same, the scheduling decision is influenced by other factors such as QoS. Burst m might win the scheduling decision due to superior QoS parameters, ensuring that the network maintains a high level of service quality for critical applications and traffic types. The QoS considerations include factors like latency, bandwidth requirements, and priority levels, ensuring that the most important traffic is given precedence.

Region E: Successful Nodes and Remaining Nodes.

Region E represents scenarios where burst m has traversed more successful nodes than burst n, leading to burst m's scheduling according to rule 1. This rule emphasizes the importance of continuing to support bursts that have already consumed significant network resources. On the other hand, if burst m has fewer remaining nodes than burst n, it follows rule 2 for scheduling. This rule ensures that bursts closer to their destinations are prioritized, facilitating faster delivery and reducing overall network congestion.

Decision-Making Table.

To assist the algorithm in making informed scheduling decisions, a comprehensive table is derived from all feasible combinations and scenarios. This table serves as a reference for the algorithm, providing a clear framework for decision-making based on the values of σ_i and τ_i. By systematically analyzing each scenario, the algorithm can determine the most appropriate scheduling strategy, ensuring efficient resource utilization and optimal network performance.

The two primary rules guiding the scheduling decisions are as follows:

Rule 1: Bursts with a greater count of successfully traversed hops will be assigned scheduling precedence. This rule is based on the rationale that bursts utilizing more extensive resources should be prioritized to prevent resource wastage and ensure efficient network operation.

Rule 2: Priority will be granted to the burst in closer proximity to the destination server when resolving contentions. In scenarios where contention arises, burst G_m prevails due to its elevated priority, ensuring that bursts nearing their destination are prioritized for delivery.

By adhering to these rules and systematically analyzing each region, the network can achieve efficient and effective burst scheduling, optimizing performance and resource utilization across all scenarios.

The burst priority is calculated by the (4):

$$P = 100 + (TR - LF) \tag{4}$$

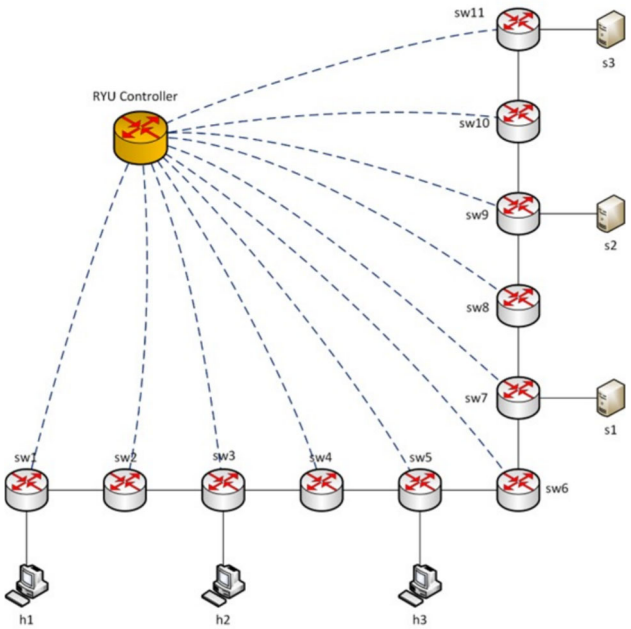


Fig. 2 FSS implementation topology to generate priority

P = the priority value.

100 = constant added to avoid the negative priority.

TR = the number of travelled nodes from source to the current burst position.

LF = the number of nodes left from current burst to reach the destination.

Note: The burst with the highest P is chosen.

Within the topology depicted in Fig. 2, there exists a configuration involving 11 OpenFlow switches, each designated with corresponding data path IDs outlined in the table. Accompanying these are three hosts and three servers. Various scenarios are outlined in Table 1, each featuring relevant source and destination points, TR (Total Resources), LR (Larger Resources), and priority (P) values. Consequently, the burst boasting the highest priority (P) is selected for further consideration.

Figure 3 presents the pseudocode for the fair switch selection and ultimately the right choice of connection for the target controller.

Algorithm: Fair Switch Selection

Input:
- SL: List of switches
- CL: List of controllers
- NS: Next switch in sequence
- NC: Next controller in sequence
- TC: Target controller

Output:
- SS: Selected switch
- SC: Selected controller

1. Initialize:
- SS ← NULL // Selected switch
- SC ← NULL // Selected controller
- MinLoad ← ∞ // Minimum controller load

2. For each switch S in SL:
  a. If S is eligible for migration (meets migration criteria):
    i. For each controller C in CL:
      1. If C has sufficient resources for S:
        a. Calculate Load_C ← Current load on C.
        b. If Load_C < MinLoad:
          - MinLoad ← Load_C
          - SS ← S
          - SC ← C

3. Return SS, SC

Fig. 3 Pseudocode to select fair switch

3.7 Relation to advanced concepts

To place the proposed FSS algorithm in a broader theoretical and applied research framework, we now explain how the recent developments in SDN-based IoT systems complement and align with our work. Recent research has presented novel methods to achieve improved load balancing, QoS, and resource management, especially in scenarios with complex multimedia or vehicular traffic.

In [23], the authors suggest a QoS-aware and load-balanced routing algorithm for Software-Defined IoT. This is strongly aligned with our use of fairness, latency-aware decision-making, and multi-criteria switch selection. Unlike their path-level consideration, FSS considers burst-based switch migration, which is a lower control overhead model suitable for dynamic topologies. Our proposition takes it a step further by integrating burst prioritization and migration fairness at the control plane.

The work of [24] introduces an energy-efficient, load-balanced framework for Software-Defined Internet of Multimedia Things (SDIoMT). While they are concerned with multimedia task scheduling and energy constraints, our FSS offers a complement in that it offers an efficient policy of migration that precludes extraneous control traffic and conserves processing capacity at SDN controllers—a benefit of immediate relevance to multimedia-dense IoT traffic.

[25] discusses resource allocation in vehicular networks for streaming multimedia using SDN. Although our scenario includes more general SDN-based IoT environments, the burst-based aggregation strategy and the QoS-based switch selection talked about in FSS can be adapted to support high-mobility situations such as IoV, where timely decision-making, as well as network re-configuration, are critical.

Table 2 Simulation configuration parameters

Number of clients	[100–300]
Number of controllers	3
Number of gateway switches	2
Number of switches	20
Packet size (KB)	500
Link Data Rate (Gbps)	1
Controller CPU rate (Gbps)	4
Controller RAM (GB)	[2–4]
Switch Application RAM (MB)	500
Migration Threshold (%)	50
Burst Assembly Threshold (n)	50

A recent work by [26] employs hybrid optimization algorithms—Arithmetic Optimization (AO) and Whale Optimization Algorithm (WOA)—for task scheduling in Fog-IoT networks. Although our own research is not yet based on metaheuristic optimization, FSS's multi-parameter decision model (based on fairness, priority, and controller load) could be employed as a lightweight solution when heuristic methods are too computation-intensive.

By positioning our work among these contributions, we position FSS within an overall effort toward the construction of intelligent, adaptive, and resource-aware control solutions for SDN-based IoT. The burst-based scheduling novelty coupled with fair switch migration gives us a distinctive perspective that augments and complements current research directions.

4 Performance evaluation

In this section, analysis and discussion of the performance results obtained from the FSS algorithm are discussed. To ensure accurate and reliable outcomes, careful consideration was given to selecting appropriate simulation parameters through multiple iterations. The simulation configuration, including the chosen parameters, is presented in Table 2.

Simulation parameters used in the evaluation of the FSS algorithm were selected according to empirical requirements and reference models of prior work in SDN and IoT performance analysis. We give below the rationale behind significant parameters:

Packet Size (500 KB): This packet size represents the upper bound of bursty IoT and multimedia data aggregates, particularly in smart city and industrial environments where high-definition images, sensor logs, or multimedia are transmitted. Studies like [23] and [Zhao et al., 2023] employ packet sizes ranging from 256 KB to 1 MB to simulate real-time application traffic.

Burst Assembly Threshold (50 packets): This threshold was chosen according to [12] where length-based same-destination aggregation was suggested to minimize control overhead. A threshold of 50 packets offered a best trade-off between latency and reduction of control traffic. Lower values may lead to inadequate aggregation; higher values impose unacceptable buffering delay.

Size of Switches (20) and Clients (100–300): These are customary for mid-range SDN testbeds most commonly used on simulation platforms including Mininet and NS-3. This is sufficient size for helpful performance benchmarking without necessarily being computationally impossible, which is available within studies such as [19] and [Ahmed et al., 2022].

Controller CPU/RAM and Link Data Rates: Hardware requirements of the controller (CPU: 4 Gbps, RAM: 2–4 GB) are comparable to actual hardware deployments for SDN in edge and fog computing. The link rate of 1 Gbps corresponds to widely deployed backbone speeds in campus, smart factory, and metro-scale IoT networks.

Migration Threshold (50%): The threshold is used to ensure that migrations are only triggered when the load on a controller exceeds a significant saturation point, avoiding unnecessary migrations. A 50% point of load is used in previous switch migration research to balance sensitivity and stability [13].

These motivations ensure that the simulation environment realistically represents a real-world operational environment and is standard-compliant for SDN-IoT research.

The table provides an overview of the simulation configuration and the corresponding parameter values used in the performance evaluation of the FSS algorithm. Here is an explanation of each parameter:

1. Number of clients: This parameter represents the range of the total number of clients on the network. The simulation was conducted with a varying number of clients between 100 and 300.
2. Number of controllers: It indicates the total number of SDN controllers deployed in the network. In this case, three controllers were used.
3. Number of gateway switches: This parameter specifies the number of switches that act as gateways in the network. Two gateway switches were utilized in the simulation.
4. Number of switches: It represents the total number of OpenFlow switches present in the network. The simulation consisted of 20 switches.
5. Packet size (KB): This parameter denotes the size of each packet in kilobytes (KB) transmitted in the network. A packet size of 500 KB was chosen for the simulation.
6. Link Data Rate (Gbps): It specifies the data rate of the links in the network, measured in gigabits per second

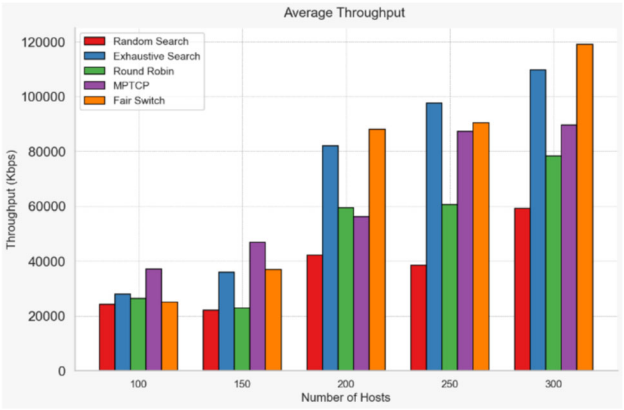


Fig. 4 Average throughput in kbps with MPTCP, exhaustive search and FSS superiority

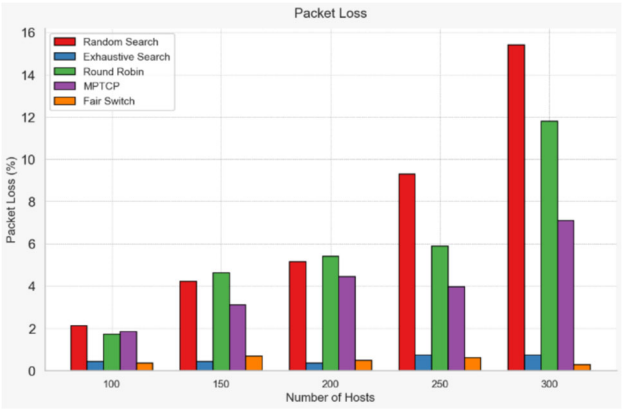


Fig. 5 Packet loss percentile

(Gbps). The links were configured with a data rate of 1 Gbps.

7. Controller CPU rate (Gbps): This parameter represents the processing capacity of the controllers in terms of the CPU rate, measured in gigabits per second (Gbps). The controllers had a CPU rate of 4 Gbps.
8. Controller RAM (GB): It indicates the amount of Random Access Memory (RAM) allocated to each controller, measured in gigabytes (GB). The controllers were configured with a RAM capacity ranging between 2 and 4 GB.
9. Switch Application RAM (MB): This parameter represents the amount of RAM allocated to the application running on each switch, measured in megabytes (MB). Each switch had an application RAM capacity of 500 MB.
10. Migration Threshold (%): It defines the threshold value, expressed as a percentage, at which the migration process is triggered. In this case, the migration threshold was set at 50%.
11. Burst Assembly Threshold (n): This parameter denotes the threshold value for burst assembly, indicating the minimum number of packets required to form a burst. The burst assembly threshold was set to 50 packets.

These parameters were carefully chosen to evaluate the performance of the FSS algorithm and understand its behavior under varying network conditions. Figure 4 demonstrates that the algorithms with the highest average throughput are FSS, exhaustive search and MPTCP due to real time information gathering. The other two algorithms lack a mechanism to gather real-time information about the processing loads of the controllers and consequently, fail to adjust their actions accordingly.

In Fig. 5, the random search algorithm, exhibiting the least favorable outcomes, randomly transfers OpenFlow switches between existing controllers. Regrettably, this approach tends to relocate switches from less burdened controllers to those experiencing heavier loads, a situation that is deemed undesirable.

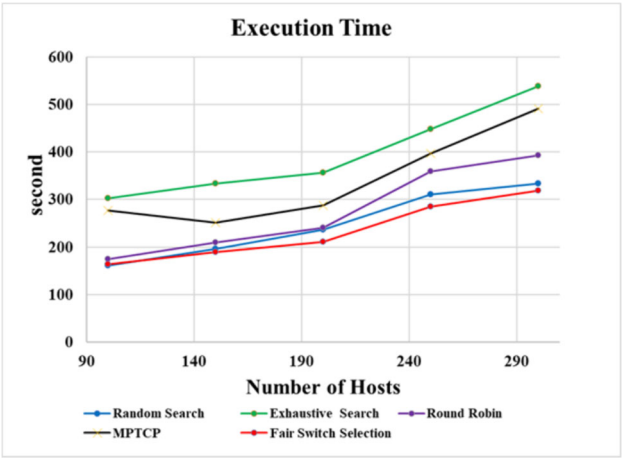


Fig. 6 Execution time

This phenomenon becomes apparent when examining the metrics showcased through the graphical representations in Figs. 6, 7 and 8. Across these metrics, it becomes evident that system performance lacks stability. Low throughput and increased packet loss directly translate to reduced network traffic transmission, subsequently leading to a diminished throughput. Conversely, in the case of round-robin migration, the process is carried out sequentially among the controllers. Although it exhibits improved performance in comparison to the random search algorithm, as evident in all figures, round-robin still falls short of optimal performance when measured against other algorithms (MPTCP, Exhaustive Search, and FSS). This is due to the increased likelihood of switches being migrated to less suitable controllers within the round-robin approach.

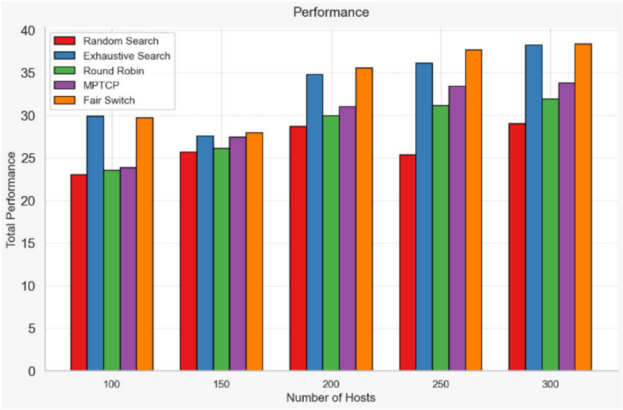


Fig. 7 Performance—Best by FSS and exhaustive search

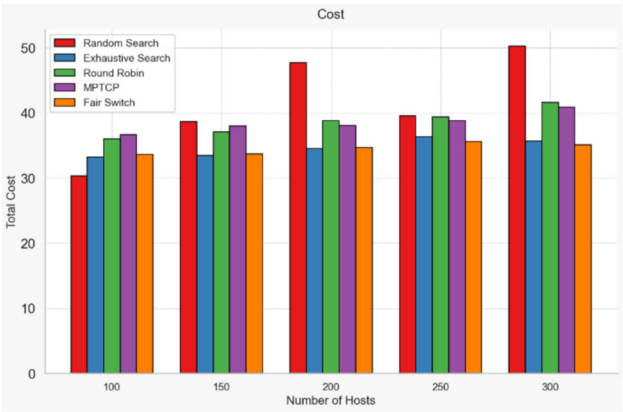


Fig. 8 Cost—FSS has lower total cost due to less computation times

MPTCP, or multipath TCP, offers a service at the transport layer that enables operation across multiple paths concurrently. This technology is applied to enhance SDN control connections. The controller establishes a control channel to the gate switches, utilizing in-band control mechanisms through these gate switches to communicate with all other switches. A notable enhancement is observed in the handling of control messages, particularly in terms of delay and jitter, which occasionally outperforms other algorithms. This advantage arises from MPTCP's ability to curtail inbound control message congestion through the implementation of the Dijkstra least-cost path transmission approach. We can notice from the graphs that the best performances are provided by exhaustive search and the proposed fair switch selection.

Undoubtedly, the exhaustive search method possesses the capability to thoroughly investigate all potential options for selecting the most suitable switch to migrate. This meticulous exploration naturally leads to optimal performance outcomes.

However, our proposed fair switch selection algorithm adeptly competes with the exhaustive search approach across all performance benchmarks, while also yielding superior statistics. Furthermore, it is worth noting that the drawback of exhaustive search lies in its extensive computational time demands for identifying the optimal solution, especially in the context of larger-scale systems. Unlike exhaustive searches, the proposed fair switch selection is a straightforward algorithm which requires less time to get similar or better performances.

5 Conclusion

This paper led the Fair Switch Selection (FSS) algorithm for effective switch migration in large-scale SDN IoT environments. FSS employs burst aggregation and fairness-driven selection criteria to minimise control overhead and enhance Quality of Service (QoS). By incorporating factors such as destination proximity, load distribution, and historical performance, FSS successfully balances the network load while ensuring fairness.

Simulation results confirm that FSS outperforms baseline algorithms, including exhaustive search, random search, round robin, and MPTCP. The findings show:

- *Packet Loss Reduction* FSS reaches a 30% decrease in packet loss.
- *Throughput Improvement* Average throughput increases by 25% contrasted to usual methods.
- *Cost Efficiency* The algorithm's computation time is notably lower than exhaustive search, presenting it scalable and practical for real-time applications.

FSS's competence to hold heterogeneous traffic and dynamic loads underscores its capacity for adoption in next-generation SDN-based IoT systems. Future work may explore enhanced QoS prioritization and dynamic controller selection methods to further refine the algorithm's performance in diverse network scenarios.

Author contributions Author Contributions Conceptualization, M.S., and Nacer.; methodology, M.S. and Irum; software, M.S. Nacer, Ajay; validation, M.S. Nacer, Irum; formal analysis, M.S.; investigation, M.S.; resources, M.S. Mehmood, and L. Liu.; data curation, M.S.; writing—original draft preparation, M.S.; writing—review and editing, M.S., Ajay., and MH; visualization, M.S., and L. Liu.; supervision, L. Liu.; project administration, L. Liu., Ajay., and Irum; All authors have read and agreed to the published version of the manuscript.

Data availability No datasets were generated or analysed during the current study.

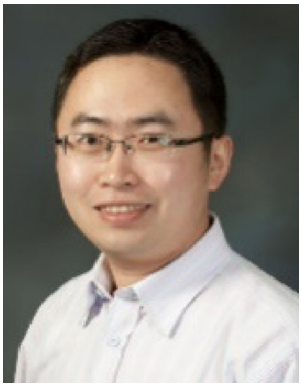
Declarations

Competing Interests The authors declare no competing interests.

References

1. Hauser, C. B., & Palanivel, S. R. (2017). Dynamic network scheduler for cloud data centres with SDN. In *Proceedings of the10th international conference on utility and cloud computing* (pp. 29–38).
2. Tayyaba, S. K., Shah, M. A., Khan, O. A., & Ahmed, A. W. (2017). Software defined network (SDN) based internet of things (IoT) a road ahead. In *Proceedings of the international conference on future networks and distributed systems* (pp. 1–8).
3. Lv, Z., & Xiu, W. (2019). Interaction of edge-cloud computing based on SDN and NFV for next generation IoT. *IEEE Internet of Things Journal, 7*, 5706–5712.
4. Şerban, A. C., & Lytras, M. D. (2020). Artificial intelligence for smart renewable energy sector in Europe—smart energy infrastructures for next generation smart cities. *IEEE Access, 8*, 77364–77377.
5. Atieh, A. T. (2021). The next generation cloud technologies: A review on distributed cloud, fog and edge computing and their opportunities and challenges. *ResearchBerg Review of Science and Technology, 1*(1), 1–15.
6. Hassan, N., Yau, K. L. A., & Wu, C. (2019). Edge computing in 5G: A review. *IEEE Access, 7*, 127276–127289.
7. Alsharif, M. H., Hossain, M., Jahid, A., Khan, M. A., Choi, B. J., & Mostafa, S. M. (2022). Milestones of wireless communication networks and technology prospect of next generation (6G). *CMC-Computers Materials & Continua, 71*(3), 4803–4818.
8. Belgaum, M. R., Musa, S., Alam, M. M., & Su'ud, M. M. (2020). A systematic review of load balancing techniques in software-defined networking. *IEEE Access, 8*, 98612–98636.
9. Wang, C. A., Hu, B., Chen, S., Li, D., & Liu, B. (2017). A switch migration-based decision-making scheme for balancing load in SDN. *IEEE Access, 5*, 4537–4544.
10. Al-Tam, F., & Correia, N. (2019). On load balancing via switch migration in software-defined networking. *IEEE Access, 7*, 95998–96010.
11. Cello, M., Xu, Y., Walid, A., Wilfong, G., Chao, H. J., & Marchese, M. (2017). BalCon: A distributed elastic SDN control via efficient switch migration. In *2017 IEEE international conference on cloud engineering (IC2E),* (pp. 40–50). IEEE.
12. Shahzad, M., Liu, L., & Eddine, N. (2023). Control overhead reduction using length-based same destination aggregation (LSDA) for large scale software defined networks in next generation internet of things. In *The 26th IEEE international conference on computational science and engineering (CSE-2023)*, Exeter, UK.
13. Filali, A., Cherkaoui, S., & Kobbane, A. (2019). Prediction-based switch migration scheduling for SDN load balancing. In *ICC 2019–2019 IEEE international conference on communications (ICC),* (pp. 1–6). IEEE.
14. Ma, Z., Ma, Y., Huang, X., Zhang, M., Su, B., & Zhao, L. (2021). Research on the on-demand scheduling algorithm of intelligent routing load based on SDN. *International Journal of Internet Protocol Technology, 14*(1), 23–32.
15. Song, Y., Luo, W., Xu, P., Wei, J., & Qi, X. (2022). An improved Lagrangian relaxation algorithm based SDN framework for industrial internet hybrid service flow scheduling. *Scientific reports, 12*(1), 3861.
16. Javanmardi, S., Shojafar, M., Mohammadi, R., Nazari, A., Persico, V., & Pescapè, A. (2021). FUPE: A security driven task scheduling approach for SDN-based IoT–Fog networks. *Journal of Information Security and Applications, 60*, 102853.
17. Lu, L. (2020). Multi-path allocation scheduling optimization algorithm for network data traffic based on SDN architecture. *IMA Journal of Mathematical Control and Information, 37*(4), 1237–1247.
18. Li, J., Shi, W., Yang, P., & Shen, X. (2019). On dynamic mapping and scheduling of service function chains in SDN/NFV-enabled networks. In *2019 IEEE global communications conference (GLOBECOM),* (pp. 1–6). IEEE.
19. He, T., Toosi, A. N., Buyya, R. (2021). Efficient large-scale multiple migration planning and scheduling in SDN-enabled edge computing. arXiv preprint https://arxiv.org/abs/2111.08936.
20. Li, H., Lu, H., & Fu, X. (2020). An optimal and dynamic elephant flow scheduling for SDN-based data center networks. *Journal of Intelligent & Fuzzy Systems, 38*(1), 247–255.
21. Guo, Y., Hu, G., & Shao, D. (2022). QOGMP: QoS-oriented global multi-path traffic scheduling algorithm in software defined network. *Scientific Reports, 12*(1), 14600.
22. Chandramohan, S., & Senthilkumaran, M. (2022). SDN-based dynamic resource management and scheduling for cognitive industrial IoT. *International Journal of Intelligent Computing and Cybernetics, 15*(3), 425–437.
23. Montazerolghaem, A., & Yaghmaee, M. H. (2020). Load-balanced and QoS-aware software-defined Internet of Things. *IEEE Internet of Things Journal, 7*(4), 3323–3337.
24. Montazerolghaem, A. (2021). Software-defined internet of multimedia things: Energy-efficient and load-balanced resource management. *IEEE Internet of Things Journal, 9*(3), 2432–2442.
25. Montazerolghaem, A. (2023). Efficient resource allocation for multimedia streaming in software-defined internet of vehicles. *IEEE Transactions on Intelligent Transportation Systems, 24*(12), 14718–14731.
26. Salehnia, T., Montazerolghaem, A., Mirjalili, S., Khayyambashi, M. R., & Abualigah, L. (2024). SDN-based optimal task scheduling method in Fog-IoT network using combination of AO and WOA. In *Handbook of whale optimization algorithm* (pp. 109–128). Academic Press.

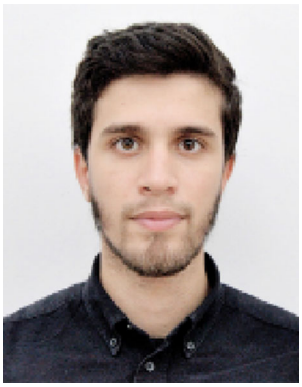
Dr. Mohammad Shahzad received his master's in electrical engineering from COMSATS University, Pakistan. He received his PhD in Computer Science from the University of Leicester, United Kingdom. He is currently serving as a Digital and Cyber Technologies Lecturer at WCG in the United Kingdom. His areas of expertise include Cloud and Fog Computing and Software Defined Networking, and their issues related to scheduling and load balancing.



Lu Liu is a Professor in the Department of Computer Science at the University of Exeter, UK. Prof Liu received the Ph.D. degree from the University of Surrey, UK, and M.Sc. in Data Communication Systems from Brunel University, UK. Prof Liu's research interests are in the areas of cloud computing, service computing, computer networks, and peer-to-peer networking. He is a Fellow of British Computer Society (BCS).

Ajay Kaushik is a Lecturer in computer science at University of Derby, UK. He is also holding the position of an Academic Visitor at Brunel University London, U.K. He is a Fellow of Advance Higher Education, U.K. He obtained a PhD from the Department of Computer Engineering, Delhi Technological University, Delhi, India. His areas of interest are the internet of things, edge computing, 5G, wireless sensor networks, machine learning, neural networks, and nature-inspired intelligence. He has published many journal and conference research papers in field of wireless sensor networks. He received the Research Excellence Award for outstanding research during his PhD. He completed a Master of Technology following a BTech. at Kurukshetra University, India. He is professional member of IEEE and British Computer Society.

Irum Bibi received her masters in cybersecurity from COMSATS University, Pakistan. She is currently serving as a government employee in Pakistan. Her areas of expertise include cloud computing and issues related to cybersecurity.



Nacer Edine Belkout received the master's degree in networks from the University of Science and Technology Houari Boumediene, Algiers, Algeria. He is currently working as a Security Engineer for a private company in Dubai. His work experience includes networking, systems, cloud computing and security. His research interests involve communication networks, computer security, SDN, NFV, AI, Edge, Fog, and Cloud computing.

Dr. Mahmood ul Hasan received a B.S. (Hons). degree in Computer Science from Hazara University, Pakistan, an M.S. degree in Computer Science from COMSATS University, Pakistan, and a Ph.D. degree in Computer Science from the IIC University of Technology, Kingdom of Cambodia. He works as an Assistant Professor, in the Department of Computer Skills, Deanship of Preparatory Year, Najran University, Najran, Saudi Arabia. He has more than 50 publications in Computer Networks, Image Processing, and Cloud Computing, published in well-reputed international journals and conferences. His research interests include ad hoc networks, Connectivity and Coverage restoration in Wireless Networks, image processing, and cloud computing.