IEEE Access
Multidisciplinary Rapid Review Open Access Journal

Received 20 August 2023, accepted 5 September 2023, date of publication 13 September 2023,
date of current version 21 September 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3314791

RESEARCH ARTICLE

Self-Socio Adaptive Reliable Particle Swarm Optimization Load Balancing in Software-Defined Networking

MOHAMMAD RIYAZ BELGAUM[1,2], SHAHRULNIZA MUSA[3,4], FUEAD ALI[3], MUHAMMAD MANSOOR ALAM[1,5], ZAINAB ALANSARI[6], (Member, IEEE), SAFEEULLAH SOOMRO[7,8], (Member, IEEE), AND MAZLIHAM MOHD SU'UD[1]

[1]Faculty of Computing and Informatics, Multimedia University, Cyberjaya 63100, Malaysia
[2]G. Pullaiah College of Engineering and Technology, Kurnool, Andhra Pradesh 518452, India
[3]Malaysian Institute of Information Technology, Universiti Kuala Lumpur, Kuala Lumpur 50250, Malaysia
[4]UniKL–LR University Joint ICT Laboratory (KLR-JIL), Universiti Kuala Lumpur, Malaysia–La Rochelle University, 17000 La Rochelle, France
[5]Faculty of Computing, Riphah International University, Islamabad 46000, Pakistan
[6]University of Technology and Applied Sciences, Muscat 133, Oman
[7]American National University, Louisville, KY 40216, USA
[8]University of Fairfax, Salem, VA 24153, USA

Corresponding author: Mazliham Mohd Su'ud (mazliham@mmu.edu.my)

ABSTRACT The ever-increasing heterogeneous connections and the demands of the users pose many new challenges to the network service providers to sustain by providing improved quality of service (QoS). Software-defined networking (SDN) is a game changer in networking by allowing user customization to enhance performance. With the advent of 5G and the increasing user requests, a massive volume of heterogeneous traffic is generated in the network, increasing load. Currently, the existing load balancing techniques lack efficiency in handling the load under unicontroller deployment. In addition, the network paths selected must also be reliable and optimal. We proposed the self-socio adaptive, reliable particle swarm optimization (SSAR-PSO) load balancing technique to address the issue of load balancing in the unicontroller deployment of SDN. In the proposed technique, the performance of the node itself, known as direct information, and the performance of the neighbouring nodes, known as indirect information, were considered to identify the reliable node to form an optimal path. Simulation results showed that the proposed technique outperforms the existing state-of-the-art techniques under TCP and UDP load in the following network performance metrics: latency, packet loss ratio, throughput, average round trip time, and bandwidth utilization ratio.

INDEX TERMS Load balancing, particle swarm optimization, quality of service, reliability, software-defined networking.

I. INTRODUCTION

Implementing new services based on the changing requirements of the users while using cloud computing, IoT, sensors, and other smart devices in building a smart environment requires reliable and efficient networking technologies. The heterogeneous network infrastructure demands the manageability of resources to attain efficiency. It needs centralized control and a view of the entire network to manage the network. The programming flexibility allows the network to support existing and future services. SDN is a solution to handle challenges with agility.

As the intelligence is transferred to the upper level, the forwarding platform depends solely on controller commands. The controller functions as a root to gather messages and issue commands, whereas switches operate in response to commands they receive. In this aspect, the transition to controller mapping substantially impacts

The associate editor coordinating the review of this manuscript and approving it for publication was Salekul Islam.

the SDN's dependability. The customized software making the device function depending on the vendor is eliminated by SDN, and load balancing improves network management. There are several load-balancing techniques to efficiently manage requests from heterogeneous machines, which can be categorized as conventional and artificial intelligence load-balancing techniques [1]. The traditional network devices ultimately depend on the functioning of the algorithms fed to them, while the SDN controller load balancing can make decisions in a centralized way. The conventional load balancing techniques are static and state oblivious. The controller efficiently handles the requests to process following the optimal path, resulting in improved QoS.

Managing load is a severe problem in traffic management of WAN (wide area network). The most effective load-balancing algorithms use proxies to direct traffic to multiple server clusters. More users mean more servers must deal with a large access request quickly, meaning the latency must be minimized. The more time a user must wait before accessing a server, the lower the quality of the service [2]. One of the crucial goals of the SDN controller is to balance the load [3]. The load is increasing as more and more enterprises move their data to the cloud, and the applications to serve the client's requests are also growing in Software Defined Networking, which makes the performance poor.

Moreover, a non-reliable link and node selection degrades the QoS [4], [5], [6]. To serve the clients and enterprises to their satisfaction, it is required to balance the load efficiently. The implementation of existing load balancing techniques in [7] shows that there is no verification for checking the reliability of the node and path used for transmitting the packets from the source host to the destination host. The packets are forwarded through the nodes to reach the destination following an optimal path. Various conventional load-balancing techniques and artificial intelligence-based load-balancing techniques currently manage the load to provide QoS to the users. Load balancing in SDN leads to discovering the best pathway and node for the fastest delivery of requests [8]. However, such load-balancing techniques have not considered reliability in following the path and selecting the node to forward the packet to the destination. Therefore, the problem of reliability while selecting the node and path during load balancing in SDN has been considered here to be addressed with our proposed research to improve the QoS.

In this paper, we contributed an adaptive meta-heuristic-based load-balancing approach that minimizes latency and enhances the QoS in SDN. The authors aim to find a reliable switch in the network to send a flow table entry based on which the packet can reach the destination. Our proposed method differs from other researchers' work because we consider the neighbouring switches' view of a current switch based on social contacts to ensure reliability, and named it self-socio adaptive, reliable particle swarm optimization (SSAR-PSO). The term adaptive considers the reliable path and node, which comes from direct information(self-node) and indirect information (neighbouring nodes). The reliability computed using the metrics from direct and indirect information is adapted in selecting the node and path to reach the destination. This research aims to provide a self socio-adaptive load balancing with reliability in a unicontroller deployment scenario. Moreover, an artificial intelligence optimization technique is also used to find the optimal path. Therefore, by integrating the AI optimization technique with the technique to find a reliable switch, our method outperforms in terms of QoS compared to the results of other techniques. This research aims to address the load-balancing issue in software-defined networking. Within the domain of the control plane, the scope is limited to the unicontroller load. Efficient load balancing in unicontroller deployment will reduce the need for a multicontroller [7], [9]. Therefore, in this paper, our significant contributions are:

- Ensuring that the request is forwarded to a reliable node on the source-to-destination path, collecting the direct and indirect information from the switches and considering them as inputs to the particle swarm optimization technique.
- The time taken to handle each request is minimized, which helps the controller to minimize the average latency.
- Improving QoS in terms of various metrics.

Following the introduction section, section II presents the background knowledge and the problem definition. Section III discusses the related works. Section IV puts forward the proposed framework. Section V constitutes a general simulation setup and presents the metrics used to measure the performance. Section VI presents the results and discussion, and section VII summarizes the discussion in the conclusion and future work.

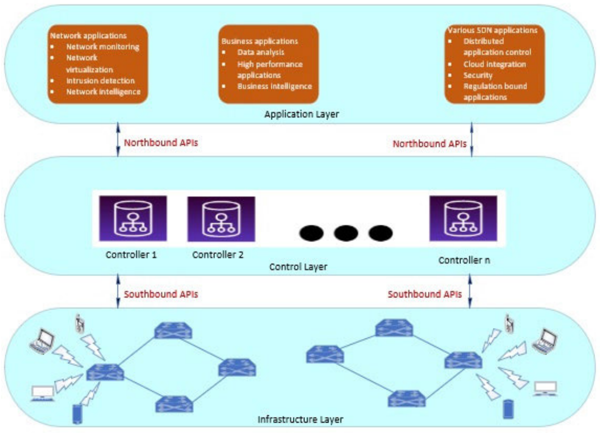


FIGURE 1. SDN architecture.

II. BACKGROUND

SDN has taken over traditional networks.

From the above figure 1, the SDN layered architecture, the controller in the control layer of SDN handles various types of traffic. And because of the users' increasing needs and services provided by the service providers, the traffic also has increased, making it a bottleneck for the controller to manage the load. There are two types of controller deployments in SDN to handle the load: unicontroller and multicontroller, as shown in figure 2 and figure 3, respectively.

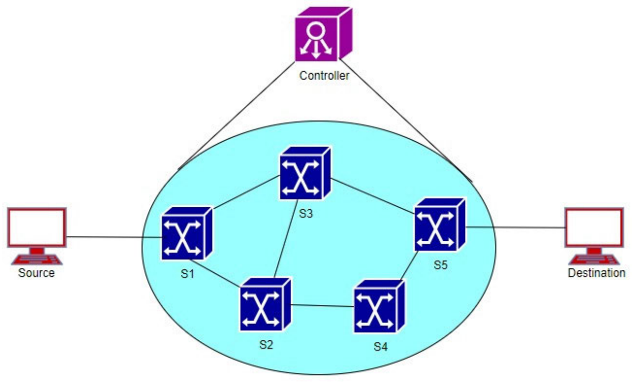


FIGURE 2. **Unicontroller deployment model.**

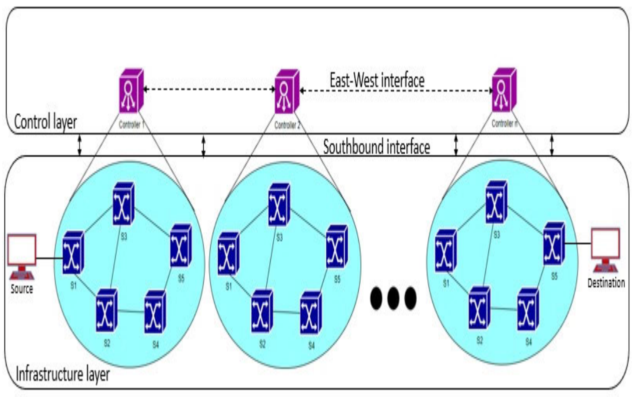


FIGURE 3. **Multicontroller deployment model.**

In a unicontroller deployment model, as shown in figure 2, limited network components are scalable and connected to the controller. However, this number depends on various parameters like the type of traffic handled, peak time, service provided, etc. When the load on the controller in the unicontroller deployment model increases, the need to migrate nodes from one cluster to another arises in a multicontroller deployment, and selecting a minimum load cluster is a different research objective.

Figure 3 presents the multicontroller deployment model, where multiple controllers communicate with each other using the east-west interface. However, if the traffic in the unicontroller deployment model is efficiently handled, then the need for multicontroller deployment will be minimized. This research focuses on the minimization of latency in handling requests by improving the QoS.

Fat tree topology is commonly used in data center environments [10], [11]. It is a K-ary tree. The number of nodes in the tree grows and shrinks based on the value of k. Fat tree topology is designed to provide high network bandwidth and low latency for interconnectivity between network devices such as servers and switches. Each core switch is connected to various pods, where a pod is a collection of aggregation switches and edge switches. Each switch has multiple paths to reach other switches or hosts in a Fat Tree topology. Each path can be assigned a unique path identifier, and the network controller can use this information to route traffic through different paths. It achieves this by providing multiple parallel paths for data to travel between devices, allowing for better load balance and redundancy.

Load-balancing techniques that use artificial intelligence to replicate past behaviour are known as outcome-based techniques [12] and employ a hybrid approach to problem-solving. Metaheuristic methods utilize artificial intelligence to address issues in the real world. Complex problems can be solved using swarm intelligence (SI). SI entails a collective investigation of how members of a local population's behaviour interact with one another. Biological systems, in particular, frequently draw inspiration from nature. Contrary to algorithms that rely on heuristics to find solutions, swarm intelligence algorithms include an evolutionary process in searching for solutions. A significant number of candidate solutions in the search area and reliable load-balancing performance are often required for swarm intelligence applications in SDN [13]. Various swarm intelligence-based algorithms like evolutionary algorithms, particle swarm optimization, differential evolution, and ant colony optimization and their variants are used to solve metaheuristics problems [14], [15]. The widely used swarm intelligence techniques for load balancing in SDN are ant colony optimization (ACO) and particle swarm optimization (PSO) [16], [17], [18].

III. RELATED WORK

The authors in [19] proposed a mechanism known as a software-defined optimal network utilization mechanism (SONUM) with a focus on minimizing the network packet loss rate. Simulative experiments were conducted on a fat tree topology with some fixed parameters to measure the performance of the proposed mechanism over ECMP and Hedera. Throughput and packet loss rate showed improved results; however, other metrics were not considered. The link bandwidth, which plays a vital role in load balancing, was given the least importance in their work.

However, the other researchers considered link bandwidth and incoming flows in their work to handle the load in SDN [12]. Identification of flow as mice flow and elephant flow was considered in proposing size-knapsack-PSO. The proposed work was implemented on fat tree topology and considered TCP traffic alone for assessing the performance. The packet loss rate and flow completion time have improved over SONUM, Hedera, and ECMP load balancing techniques.

The researchers in [20] have handled the load balancing issue in cloud computing by fine-tuning the inertia weight in the PSO technique. The tuning of inertia weight was done to have a balance between local and global searches. The proposed approach lowered the makespan and improved the throughput and average resource utilization ratio. However, the other QoS metrics were not considered in assessing the proposed approach. Minimizing the response time is the challenge considered by the authors in [9]. An efficient switch-controller pair reduces the throughput with more response time. The genetic algorithm with PSO used in their work assigned weights to the particles and minimized the switch migrations. SDN-based topology in unmanned aerial vehicles (UAV) uses PSO to utilize a greater number of connected nodes to avoid collisions and use alternate routes during the failure of routes in [21]. Packets delivered increased with less delay, but the nodes selected to transfer the packets were not verified to be reliable or not. The authors in [22] proposed an energy-efficient service placement technique in fog computing by minimizing the response time of latency-sensitive applications. The application deadlines, application makespan time, resources of the application module, and fog node capacities were considered to dynamically place the services. The network delay and bandwidth consumption were considered, which impacts other QoS metrics.

A comparative analysis of round robin, ACO, and improved ACO (Im-ACO) was made by the author in [23]. The author considered the weights with the node having a minimum connection, and the optimal path was selected to transfer the packets, giving an increased throughput. Most of the authors focused on minimizing the load of the overloaded controller. In contrast, the authors in [24] tried to solve the problem by distributing the load equally, in the long run, using artificial bee colony (ABC) optimization and reducing the number of switch migrations, thereby improving the performance. The authors in [25] used the ACO technique for traffic load balancing in SDN. In unicontroller deployment, the traffic was injected into the network using iperf3, and the shortest path between the source and destination pair was determined. Also, the implemented technique was assessed in terms of throughput, data transfer rate, bandwidth, and average delay. Though there was an improvement in all the metrics considered in their work, the reliable path was left undetermined. ACO and PSO techniques are compared with round-robin in the context of load balancing in SDN () [26]. Comparison in terms of parameters used, advantages, and disadvantages while balancing the load in SDN in a unicontroller deployment reveals that both techniques produce better results than the round-robin technique. However, reliability was not considered for node and path.

There was less consideration paid to using AI in SDN for load balancing. The node and path used to transfer the packets should be reliable to develop a reliable load-balancing technique.

IV. METHODOLOGY

This section presents the normative action approach used to improve existing load-balancing techniques' performance. The methodology laid down by action research is exceedingly helpful where finding existing methods and their execution is required, with the goal that its impact on the framework could be examined for further enhancements. This study finds the best strategy for a reliable node and path to reaching the destination, thereby increasing the capacity to handle more load and providing implementation to measure the framework's performance. Moreover, the normative action approach is fitted best for research that presents an innovative procedure to cope with a problem and then compares it with existing solutions to measure its effectiveness. Domain analysis in the present study includes the switch's control plane to analyze the reliable node and path to forward the packet. The controller considers direct and indirect information about the switch in the proposed model and utilizes them to identify the reliable node and path to forward the packets. The direct and indirect information helps propose, design and implement a new framework for load balancing in SDN. To determine the effectiveness of the proposed technique, implementation is to be presented that is evaluated through testing and comparison with existing techniques like round-robin, ACO, and PSO load balancing techniques to ensure that the QoS metrics are improved. This research presents proof of concept implementation regarding the load-balancing approaches in SDN.

The three basic load balancing techniques, namely round-robin, ACO, and PSO, are implemented initially. Later, our proposed framework was included in the original PSO load-balancing method. The reason for considering the round-robin technique for comparison is that it is a widely used load-balancing technique particularly used by CISCO [27]. The finest swarm optimization strategies for dealing with meta-heuristic issues are ACO and PSO, which is why they were chosen [28], [29], [30].

The initial PSO parameters are directly connected to the convergence behaviour of the optimal solution. On the other hand, PSO parameters are largely employed to provide the SDN controller with optimum control. By tuning the parameters, switches may reach their intended top speeds and accelerations, frequently equal to the network's maximum performance. A further increase in these parameters will not lead to quicker convergence, but it might result in poor control of the switches if the SDN controller and the switches are out of sync. Switches with greater maximum velocity and acceleration are needed to achieve quicker convergence. It is still possible for the researcher to exert some control over the swarm's exploration and exploitation inclinations via the adjustment of c1 and c2 [12]. The controller in SDN is the element that implements the technique to find the optimal path between source and destination nodes.

Moreover, the nodes and paths used to reach the destination are found to be non-reliable, reducing the QoS. In other words, if a better node and path is more reliable than the current one, then a modification to selecting such a reliable node and path improves the QoS. Therefore, to select a more reliable node and path, the PSO technique is modified with the integration of the proposed framework. In the techniques used in this research for performing load balancing, the node selected and the link used on the solution path is not verified as reliable. Moreover, the above metrics discussed show that they can be enhanced more by modifying the PSO technique. The reasons for selecting PSO for modification are as follows:

- It is evident from the implementation done in the authors' previous related research work that the three techniques used for load-balancing PSO have better results [7].
- PSO has a fast convergence of global search
- The metrics used to assess the performance of the load balancing technique can be used as input parameters that can exactly fit the PSO technique and can be verified for different topologies, too.

As a result, a framework with a modification to the PSO approach is provided that uses the metrics collected from the previous flows as input parameters to pick a reliable node and reliable connection on the best route from the direct and indirect sources. The proposed framework is integrated with the PSO load balancing technique by modifying the cognitive constraint by taking into account the direct information from the node and the social constraint by taking into account the indirect information, i.e., the information from the initial node's neighbouring nodes, to ensure that the node and path chosen are of maximum reliability. Self-Socio Adaptive Reliable Particle Swarm Optimization (SSAR-PSO) is the name given to the framework.

SELF-SOCIO ADAPTIVE RELIABLE PARTICLE SWARM OPTIMIZATION:

To solve the upper-level issue, PSO aims to allocate each flow to the route with the lowest cost. Each solution is written as an ordered vector of nodes (network switches) travelling from the source to the flow's destination. The particle swarm optimization algorithm is a technique for finding the best solution in the solution space by having each particle in the population follow the current superior particle at a set speed. The fitness function is determined to get the best optimal solution. For each particle, the fitness function is evaluated. In order to evaluate the fitness function of SSAR-PSO, shown in Eq. 1, the factors like packet loss ratio, delay, throughput rendered by the switch, and bandwidth utilization are considered to evaluate the reliability constraint (R) for the shortest path, which is used in the fitness function. The range of R falls between $[0, 1]$. The node and path of maximum $f(p)$ is selected.

$$f(p) = \max(R \min(path)) \tag{1}$$

The selection of PSO parameters significantly impacts the algorithm's performance and efficiency. In SSAR-PSO, the tuning of parameters is done during the updation of the particle's velocity by considering the packet loss ratio, delay, throughput, and bandwidth utilization of the switch. The parameters in the velocity update equation of the PSO algorithm are inertia weight (ω), random numbers r1, r2 and the acceleration coefficients c1, c2, which are shown in below

TABLE 1. Notations used in the proposed framework.

Notation	Description
V	Network node
E	Network link
N	Number of switches
$p_{u,v}$	Shortest path between node u to node v
r_l	Reliability of link l
r_n	Reliability of node n
S_{mj}	j^{th} switch controlled by controller m
S_{mj+1}	$(j+1)^{th}$ switch controlled by controller m
$S_{m(j+1)'}$	$(j+1)'$ are all neighbour switches of $(j+1)^{th}$ switch

Eq. 2. [31]

$$\mathrm{V}^{t+1} = (\omega \mathrm{V}^{t} + c1.r1\left(\text{pbest}-\text{pos}^{t}\right) + c2.r2\left(\text{gbest} - \text{pos}^{t}\right)) \tag{2}$$

Using Eq. 2, the position of the particle is updated following Eq. 3.

$$pos^{t+1} = pos^{t} + V^{t+1} \tag{3}$$

The three terms combinedly designate a purpose to update the velocity of the particle. The first term is the velocity at t^{th} iteration, the second term represents the cognition model, and the third is the social model. The switch's direct and indirect information are considered cognitive and socio constraints in the velocity updation in Eq. 2, the value of which impacts forwarding the packet to the next successive switch. The requests are sent to a reliable node along the reliable path to decrease latency. Because of the decreased latency, the controller can manage a greater workload.

The notations used in this section are summarised in Table 1.

Let the switches controlled by controller m be $S_m = \{s_{m1}, s_{m2}, \ldots\}$. The reliability between the source and destination nodes $R_{s,d}$ can be expressed, as shown in Eq. 4, for the optimal path.

$$R_{s,\mathrm{d}} = \prod\nolimits_{l \varepsilon E} r_l X_{l,p_{u,v}} * \prod\nolimits_{n \varepsilon V} r_n X_{n,p_{u,v}} \tag{4}$$

If node n, and link l are more reliable than other nodes and links, then they may be included in the chosen route between $p_{u,v}$. Metrics such as the packet loss ratio (PLR), transmission delay (TD), and throughput (TH) may be used to determine the node's reliability (r_n). The ratio of bandwidth used to the total bandwidth available is a useful metric for gauging the stability of the connection known as bandwidth utilization ratio (BUR) is used for link's reliability (r_l). Prior traffic data are used to compile the values for these metrics. To calculate the reliability of the whole route from the source node to the destination node, we use Eq. 4, where the first expression is the product of all reliable connections on the ideal path and the second expression is the product of all reliable nodes on the optimal path. When a packet is sent from the switch S_{mj} to S_{mj+1}, S_{mj} takes in two pieces of data about the previous traffic flow. (D), is for direct information, (ID) for information that is not directly available to the reader. As demonstrated in Eq. 5, the reliability of a node, r_n, is equal to the aggregate value of S_{mj} multiplied by S_{mj+1}.

$$r_n = \frac{1}{2}\sum(D, ID) \tag{5}$$

Under direct information, the switch S_{mj} gets information from S_{mj+1} about its packet loss ratio, delay, and throughput using Eq. 6. And under indirect information, S_{mj} gets information about S_{mj+1} from $S_{m(j+1)'}$ in terms of the packet loss ratio, delay, and throughput using Eq. 7.

$$D = S_{m(j+1)_{PLR}} * S_{m(j+1)_{TD}} * S_{m(j+1)_{TH}} \tag{6}$$

$$ID = \frac{1}{n}\sum_{j=1}^{n}(S_{m(j+1)'_{PLR}} * S_{m(j+1)'_{TD}} * S_{m(j+1)'_{TH}} \tag{7}$$

In order to obtain the indirect information, an adjacency matrix [32] is used to identify the neighbouring switches of each switch S_{mj}. Wherever there is a link from one switch to other, it is represented as 1 and when there is no link between the switches then it is represented as 0. For all the links between S_{mj+1} to $S_{m(j+1)'}$ if there is link then it is represented as 1 and otherwise it is 0. In the same way, the link's reliability (r_l) can be calculated as an aggregate of using direct and indirect information of S_{mj+1} in terms of bandwidth utilization using Eq. 8.

$$r_l = \frac{1}{2}\sum(D', ID') \tag{8}$$

Under direct information, the switch S_{mj} gets information from S_{mj+1} about its bandwidth utilization using Eq. 9.

$$D' = S_{m(j+1)_{BUR}} \tag{9}$$

And under indirect information, S_{mj} gets information about S_{mj+1} from $S_{m(j+1)'}$ in terms of bandwidth utilization using Eq. 10.

$$ID' = \frac{1}{n}\sum_{j=1}^{n}(S_{m(j+1)'_{BUR}} \tag{10}$$

An aggregate of direct and indirect information of S_{mj+1} gives the reliability value to S_{mj} as the reliability of the link as r_l, as shown in Eq. 8. The index function: $X_{l,p_{u,v}} = \{0, 1\}$, $X_{n,p_{u,v}} = \{0, 1\}$ is used to verify whether the link l and node n, are part of the path or not. The problem of optimization also helps to solve the location of the controller to maximize the average network reliability between the controller and switches. The following changes are done in the updation of velocity in Eq. 2 of SSAR-PSO by evaluating the constraints mentioned in Eq. 11 to prove that it is self-socio adaptive reliable.

$$\text{c1} = D * D' \quad \text{and} \quad \text{c2} = ID * ID' \tag{11}$$

TABLE 2. Simulation parameters.

Parameter	Description
Simulator	Mininet 2.2.1 [34], [35]
Controller	Floodlight
Topology	Fat tree
Switch	OVS (Open Virtual Switch) [36], [37]
Protocol	OpenFlow V1.3
Traffic Generator	D-ITG
Types of load	TCP and UDP
Packet size	1024 bytes

where c1 is a cognitive constraint, and c2 is socio-constraint. The flowchart in below figure 4, shows the modifications done to PSO to use SSAR-PSO.

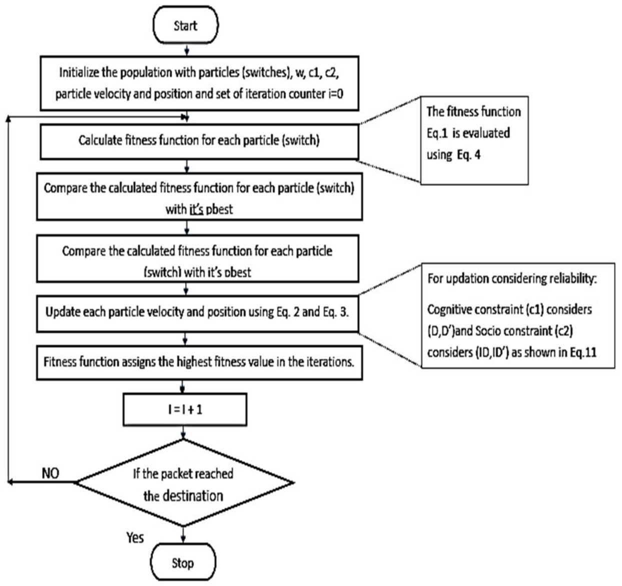

Start
Initialize the population with particles (switches), w, c1, c2, particle velocity and position and set of iteration counter i=0
Calculate fitness function for each particle (switch)
The fitness function Eq.1 is evaluated using Eq. 4
Compare the calculated fitness function for each particle (switch) with it's pbest
Compare the calculated fitness function for each particle (switch) with it's gbest
Update each particle velocity and position using Eq. 2 and Eq. 3.
For updation considering reliability:
Cognitive constraint (c1) considers (D,D')and Socio constraint (c2) considers (ID,ID') as shown in Eq.11
Fitness function assigns the highest fitness value in the iterations.
I=I+1
If the packet reached the destination
NO
Yes
Stop

FIGURE 4. SSAR-PSO flowchart.

Therefore, packets may be routed to the reliable node after it has been determined using the direct and indirect information outlined above. The delay is decreased because more packets may be sent at full speed to the destination when they are forwarded to a reliable node. Reduced latency improves the controller's capacity for load balancing and its performance in terms of QoS measures [33].

V. SIMULATION SETUP AND METRICS USED

This section presents the simulation parameters considered for implementation in table 2, and the following discussion presents the metrics used to assess the load-balancing techniques.

Considering the above parameters mentioned, the inference has been made on the default packet size of 1024 bytes. However, the number of packets has been kept varying to 10 packets each time in case 1 and by sending 100 packets each time to measure the performance of each load balancing technique. So, this varying number of packets can be considered for different types of networks.

A. BANDWIDTH UTILIZATION RATIO (BUR)

The bandwidth utilization ratio (BUR) is an important assessment criterion in network transmission since it can be used to determine the load status of a single link. When $BUR = 1$, it means that the link capacity has been utilized to the maximum, and this connection may have a full load with a significant likelihood of link congestion occurring. When calculating the link bandwidth usage ratio, the SDN controller must collect cumulative transmitted bytes B_T at the relevant OpenFlow switches port to calculate the ratio. By calculating the subtraction between B_T and the cumulative sent bytes of the previous period, B_{T-1} we can get the number of data bytes transferred in this period, corresponding to the amount of bandwidth the connection uses. Finally, we divide the total bandwidth used by the connection in this timeframe by the maximum bandwidth available. Consequently, the bandwidth utilization ratio may be computed using Eq. 12.

$$BUR_{link} = \frac{B_T - B_{T-1}}{B_{max}} \tag{12}$$

B. PACKET LOSS RATIO (PLR)

Packet loss happens during the packet processing phase in a switch. Switches may lose a packet if they are overloaded with other traffic and cannot handle the incoming packets effectively. Ping time shows the busy state of the switch, whereas the packet loss ratio indicates the load condition of the route. On OpenFlow switches ports that correspond to the SDN controller, the cumulative number of transmitted packets $Packet_T$ and the cumulative number of received packets $Packet_R$ may be collected by the SDN controller. As a result, Eq. 5.13 may be used to determine the packet loss ratio of a switch on a respective path:

$$PLR = \frac{Packet_T - Packet_R}{Packet_T} \tag{13}$$

C. TRANSMISSION DELAY (TD)

The time a packet takes to move from one node to another is called delay. Transmission delay depends on the switch's performance, packet size and the state of congestion in the transmission queue. The transmission delay may show the connection's congestion state and the switch's load condition. It can be derived as shown in Eq. 14.

$$TD = \frac{Length_{packet}}{Link_{bandwidth}} \tag{14}$$

D. THROUGHPUT

It is a measure of the successful transmission of packets from one node to another node. Therefore, the throughput can be calculated using Eq. 15.

$$TH = \frac{\sum_{i=1}^{n} Packets_{received}}{\sum_{i=1}^{n} Packets_{originated}} \tag{15}$$

E. LATENCY

The time it takes for a packet to go from source to destination across a network is called the end-to-end delay. The transmission, propagation, and switch delays that occur when forwarding the packet are included in this time calculation, as shown in Eq. 16.

$$L = TD_{sr} + PD + TD_{dt} + SD \tag{16}$$

where L is the latency, TD_{sr} is the transmission delay at the source, PD is the propagation delay, TD_{dt} is the transmission delay at the destination, and SD is the delay at the switch.

Average Round Trip Time (RTT): Transmission of packets in a source and destination path involves many nodes; thus, the amount of time for a packet to travel from one node to another and back is known as round-trip time.

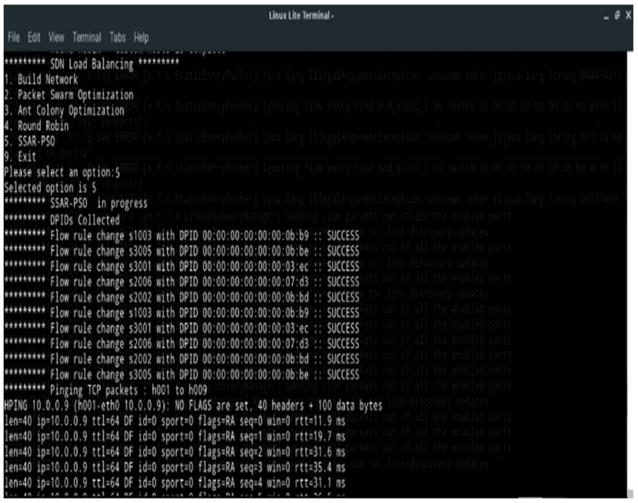
By utilizing the fat tree topology with k=4, where there are four core switches, eight aggregation switches, and eight edge switches, having 16 hosts connected and the testbed setup, the traffic is injected into the network by using D-ITG by randomly selecting the source and destination nodes. The load balancing techniques, namely PSO, ACO, RR, and SSAR-PSO, are implemented on the same pair of source and destination nodes by varying the traffic as TCP and UDP. The results are extracted and used for comparative analysis in terms of metrics like latency, packet loss ratio, average round trip time, throughput, and bandwidth utilization.

VI. RESULTS AND DISCUSSION

The performance of the proposed scheme (SSAR-PSO) is compared with the state-of-the-art schemes of load balancing in SDN. The four load-balancing techniques are implemented, and their results are compared under TCP and UDP traffic separately. The experiments have been conducted and are categorized into two cases. In case (1) each time, the packets with an increment of 10 packets upto 100 packets under both TCP and UDP traffic, and in Case (2) each time, the packets with an increment of 100 upto 1000 packets under both TCP and UDP traffic are sent from any source to destination pair. As discussed in [7], the three load balancing techniques, round robin, ACO, and PSO, are implemented and based on the functionality of the techniques used; the packets are sent to the destination host according to the flow rules. Considering the PSO technique, a reliable node and path is selected using SSAR-PSO. The packets sent from source (H001) to destination (H009) under SSAR-PSO is shown in figure 5 below.

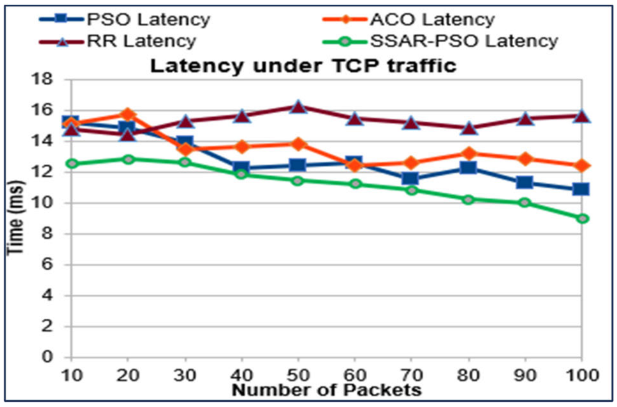
Case (1): In this case, first, the TCP traffic and, later, UDP traffic are sent separately with an increment of 10 packets each time, and the results are collected and graphically plotted.

The results in figure 6(a) show that the latency of SSAR-PSO is less than all the other load-balancing techniques used here for comparison. The average latency of SSAR-PSO is 11.070% less than that of PSO, 17.624% less than ACO, and 25.982% less than that of the RR technique, respectively, under TCP traffic. While under UDP traffic, the results

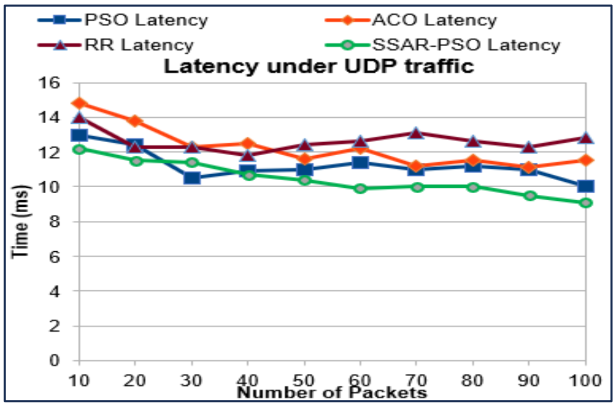


```
********* SDN Load Balancing *********
1. Build Network
2. Packet Swarm Optimization
3. Ant Colony Optimization
4. Round Robin
5. SSAR-PSO
9. Exit
Please select an option:5
Selected option is 5
********* SSAR-PSO  in progress
********* DPIDs Collected
********* Flow rule change s1003 with DPID 00:00:00:00:00:00:0b:b9 :: SUCCESS
********* Flow rule change s3005 with DPID 00:00:00:00:00:00:0b:be :: SUCCESS
********* Flow rule change s3001 with DPID 00:00:00:00:00:00:03:ec :: SUCCESS
********* Flow rule change s2006 with DPID 00:00:00:00:00:00:07:d3 :: SUCCESS
********* Flow rule change s2002 with DPID 00:00:00:00:00:00:0b:bd :: SUCCESS
********* Flow rule change s1003 with DPID 00:00:00:00:00:00:0b:b9 :: SUCCESS
********* Flow rule change s3001 with DPID 00:00:00:00:00:00:03:ec :: SUCCESS
********* Flow rule change s2006 with DPID 00:00:00:00:00:00:07:d3 :: SUCCESS
********* Flow rule change s2002 with DPID 00:00:00:00:00:00:0b:bd :: SUCCESS
********* Flow rule change s3005 with DPID 00:00:00:00:00:00:0b:be :: SUCCESS
********* Pinging TCP packets : h001 to h009
HPING 10.0.0.9 (h001-eth0 10.0.0.9): NO FLAGS are set, 40 headers + 100 data bytes
len=40 ip=10.0.0.9 ttl=64 DF id=0 sport=0 flags=RA seq=0 win=0 rtt=11.9 ms
len=40 ip=10.0.0.9 ttl=64 DF id=0 sport=0 flags=RA seq=1 win=0 rtt=19.7 ms
len=40 ip=10.0.0.9 ttl=64 DF id=0 sport=0 flags=RA seq=2 win=0 rtt=31.6 ms
len=40 ip=10.0.0.9 ttl=64 DF id=0 sport=0 flags=RA seq=3 win=0 rtt=35.4 ms
len=40 ip=10.0.0.9 ttl=64 DF id=0 sport=0 flags=RA seq=4 win=0 rtt=31.1 ms
```

FIGURE 5. Transmission of packets under SSAR-PSO.



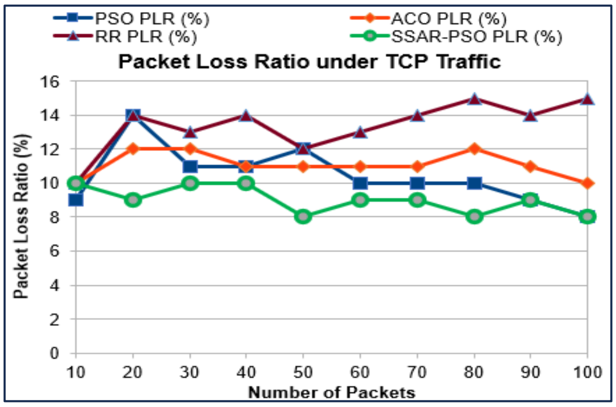
(a)



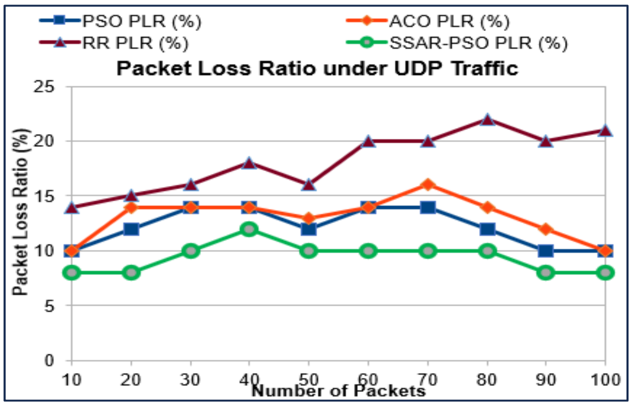
(b)

FIGURE 6. (a) : Analysis of Latency under TCP traffic with an increment of 10 packets each time. (b): Analysis of Latency under UDP traffic with an increment of 10 packets each time.

in figure 6(b) the average latency of SSAR-PSO shows a reduction of 6.770% over PSO, 14.420% reduction over ACO and 16.950% reduction over RR technique. It is evident from the above graph that as the number of packets transmitted increases, the latency has been reduced as the path and node used for transmission are more reliable than the node and path used in other techniques. Also, the efficiency of SSAR-PSO was measured in terms of metrics like packet loss ratio, throughput, average round trip time and bandwidth utilization ratio.



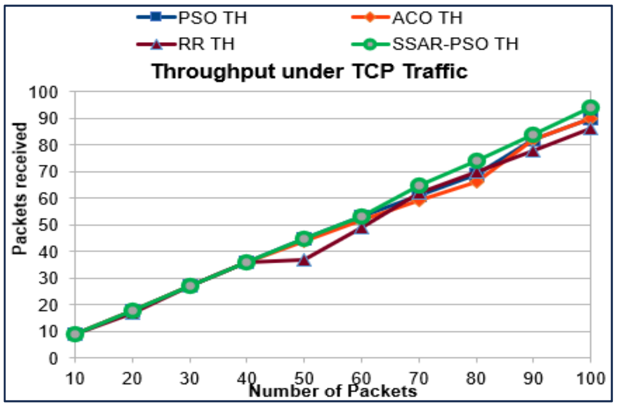
(a)



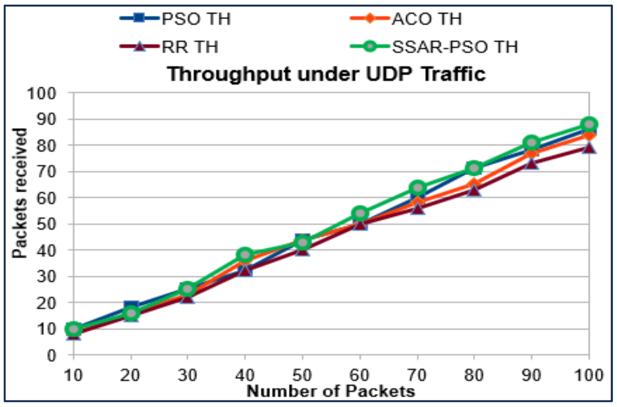
(b)

FIGURE 7. (a) : Analysis of packet loss ratio under TCP traffic with an increment of 10 packets each time. (b): Analysis of packet loss ratio under UDP traffic with an increment of 10 packets each time.

The packet loss ratio of SSAR-PSO is less than all the other load balancing techniques used here for comparison, which can be seen in figures 7(a) and 7(b). The packet loss ratio of SSAR-PSO is 11.61% less than PSO, 18.59% less than that of ACO, and 31.62% less than that of the RR technique respectively under TCP traffic. From figure 7(b), it is evident that under UDP traffic, SSAR-PSO has 22.67% less than that of PSO, 32.285% less than that of ACO and 47.43% less than that of the RR technique, respectively. The reason for this diminishing packet loss in SSAR-PSO is the selection of reliable nodes for the transmission of packets. The node selected for transmission uses direct information, i.e., from the node itself, and indirect information, i.e., from the source node's neighboring nodes. So collectively, this information, along with other inputs, is used to select a reliable node. Therefore, the packet loss of the node selected for the transmission of packets is diminishing.

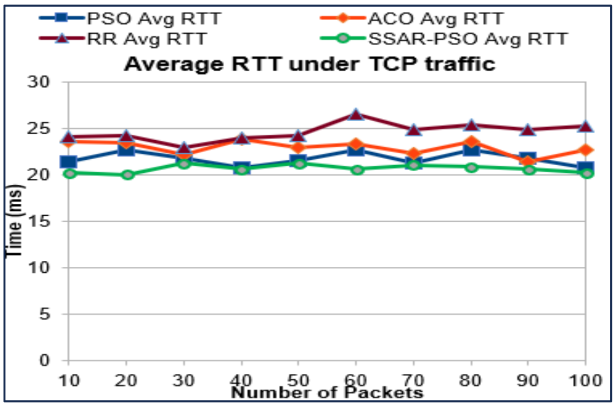


(a)

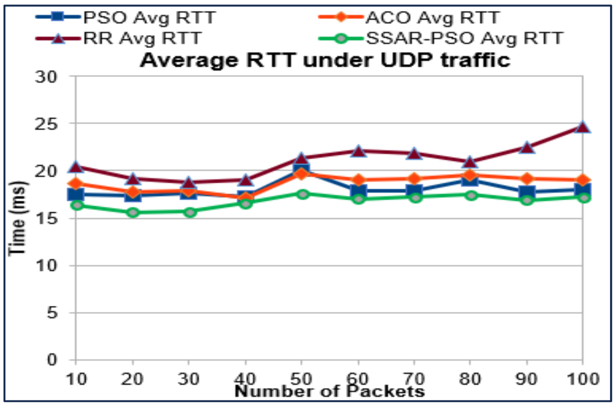


(b)

FIGURE 8. (a) : Analysis of throughput under TCP traffic with an increment of 10 packets each time. (b): Analysis of throughput under UDP traffic with an increment of 10 packets each time.



(a)



(b)

FIGURE 9. (a) : Analysis of average round trip time under TCP traffic with an increment of 10 packets each time. (b): Analysis of average round trip time under UDP traffic with an increment of 10 packets each time.

The throughput of all the load-balancing techniques is compared and the results are presented in the following figures 8(a) and 8(b).

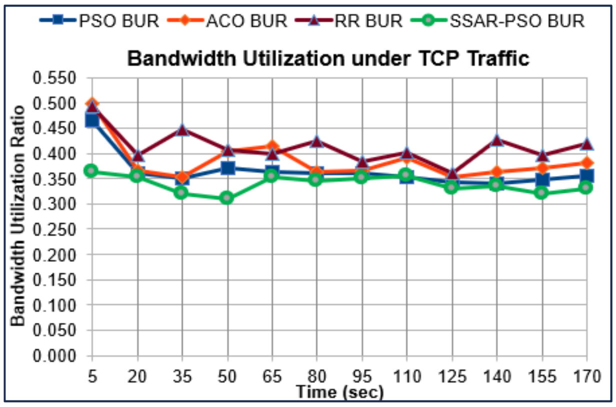
The throughput of SSAR-PSO is more than all the other load-balancing techniques used here for comparison. The throughput of SSAR-PSO is 2.07% more than that of PSO, 3.34% more than that of ACO and 6.32% more than that of the RR technique, respectively under TCP traffic. Under UDP traffic, the throughput of SSAR-PSO is 2.62% more than PSO, 7.45% more than ACO, and 12.89% more than that of the RR technique, respectively. It is evident from the above graph that the reason for this increase in the throughput of SSAR-PSO is the selection of reliable nodes for the transmission of packets. Using direct information and indirect information, the maximum metrics values are used to find the reliable node. Therefore, the throughput of each node selected for the transmission of packets is increasing.

The average round trip time incurred by implementing each of the techniques under TCP and UDP load is collected and is depicted in figures 9(a) and 9(b) respectively.

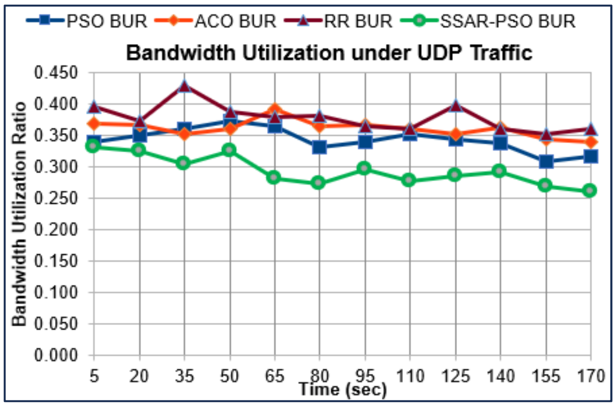
Figure 9(a) shows that the average round trip time of SSAR-PSO is less than all the other load balancing techniques used here for comparison. Under TCP traffic the average round trip time of SSAR-PSO is 4.78% less than that of PSO, 9.79% less than that of ACO and 15.93% less than that of the RR technique, respectively. Figure 9(b) shows that the average RTT of SSAR-PSO is 6.91% less than PSO, 10.22% less than ACO, and 20.09% less than that of RR respectively under UDP. Similarly, the bandwidth utilization in SSAR-PSO is less than all the other load-balancing techniques used here for comparison, under TCP and UDP traffic which can be seen in figures 10(a) and 10(b) respectively.

The average bandwidth utilization of SSAR-PSO is 6.52% less than that of PSO, 11.34% less than ACO, and 17.49% less than the RR technique, respectively, under TCP traffic. And under UDP traffic, SSAR-PSO utilizes 14.61% less than PSO, 18.74% less than ACO, and 22.50% less than RR technique, respectively. The reduced bandwidth can be used to transmit more packets.

Case (2): The same experiments are repeated under the TCP traffic and UDP traffic with an increment of 100 packets each time with different source and destination pairs. The results are collected, and comparative summarized results are presented in table 3.



(a)



(b)

FIGURE 10. (a) : Analysis of bandwidth utilization ratio under TCP traffic with an increment of 10 packets each time. (b): Analysis of bandwidth utilization ratio under UDP traffic with an increment of 10 packets each time.

TABLE 3. Comparative results with increment of 100 packets each.

Metric	Improved / Reduced	Traffic	H001 to H009			H006 to H014		
			SSAR-PSO over PSO (%)	SSAR-PSO over ACO (%)	SSAR-PSO over RR (%)	SSAR-PSO over PSO (%)	SSAR-PSO over ACO (%)	SSAR-PSO over RR (%)
Latency	Reduced	TCP	4.62	8.56	11.54	4.25	8.99	12.22
		UDP	3.61	9.52	10.86	4.12	8.55	10.55
Packet loss ratio	Reduced	TCP	4.59	10.66	13.56	4.45	9.85	12.88
		UDP	8.62	12.51	21.64	9.55	12.85	20.8
Throughput	Improved	TCP	1.1	1.88	3.72	1.25	2.15	2.85
		UDP	1.58	3.65	5.64	1.4	3.55	5.55
Average RTT	Reduced	TCP	2.86	4.25	6.86	2.58	4.1	7.2
		UDP	3.88	6.46	9.82	3.66	6.24	9.5
Bandwidth utilization	Reduced	TCP	4.82	6.74	8.85	4.56	7.12	8.55
		UDP	6.72	10.12	12.38	6.88	10.25	10.55

From the above table, it is evident that there is a reduction in latency of SSAR-PSO both under TCP load and UDP load when the packets were taken in increments of 100 each time. However, the latency reduction is not as significant as in the previous case. This explains that when more and more packets are sent, the reduction in latency is minimized because of the congestion. The controller receives many packet_in messages for which the time taken to process each such incoming message is increased.

VII. CONCLUSION AND FUTURE WORK

A modification to the PSO load balancing framework named SSAR-PSO was proposed by taking the cognitive constraint with parameters packet loss ratio, delay, throughput, and bandwidth utilization as direct information and socio-constraint with the same parameters as indirect information to find a reliable node and path to reach the destination node. The algorithm selects a reliable node based on the reliability evaluation done from the flow rules of the switches during previous transmissions. During the evaluation, it was realized that SSAR-PSO handles the load efficiently under TCP and UDP traffic, with packets sent in an increment of 10 and 100 each time. In comparison, SSAR-PSO shows better improvement in terms of latency with an incremental 10 packets each time as 26% approximately. And with an incremental 100 packets each time as 12% approximately under TCP traffic. While under UDP traffic, when packets are sent with an increment of 10 packets each time, the latency has reduced by 18% approximately and with an increment of 100 packets each time, the latency is reduced by 10% approximately. This reduced latency serves more requests, improving the load-balancing capacity. But when the number of packets sent kept increasing, like an increment of 1000 each time, then this improvement deteriorated. The reason for this reduction is due to the congestion occurring.

The simulations to compare our proposed technique with other techniques can also be conducted on other topologies, which is our future work. This technique can also be extended to multi-controller deployment, where several controllers manage the switches under them, where scalability becomes a bottleneck in unicontroller deployment. Operators are still working to develop a stable core network architecture as 5G deployments start to roll out across networks. SDN and NFV are two technologies that may be integral parts in creating such dependability. The proposed framework can be a powerful tool for reducing bottlenecks and ensuring fair traffic distribution in the next 5G networks.

REFERENCES

[1] M. R. Belgaum, S. Musa, M. M. Alam, and M. M. Su'ud, "A systematic review of load balancing techniques in software-defined networking," *IEEE Access*, vol. 8, pp. 98612–98636, 2020.

[2] S. D. A. Shah, M. A. Gregory, S. Li, R. D. R. Fontes, and L. Hou, "SDN-based service mobility management in MEC-enabled 5G and beyond vehicular networks," *IEEE Internet Things J.*, vol. 9, no. 15, pp. 13425–13442, Aug. 2022.

[3] H. Sufiev, Y. Haddad, L. Barenboim, and J. Soler, "Dynamic SDN controller load balancing," *Future Internet*, vol. 11, no. 3, p. 75, Mar. 2019.

[4] P. P. Ray, "A survey on cognitive packet networks: Taxonomy, state-of-the-art, recurrent neural networks, and QoS metrics," *J. King Saud Univ. Comput. Inf. Sci.*, vol. 34, no. 8, pp. 5663–5683, Sep. 2022.

[5] W. Rafique, L. Qi, I. Yaqoob, M. Imran, R. U. Rasool, and W. Dou, "Complementing IoT services through software defined networking and edge computing: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 3, pp. 1761–1804, 3rd Quart., 2020.

[6] G. Fernandes, J. J. P. C. Rodrigues, L. F. Carvalho, J. F. Al-Muhtadi, and M. L. Proença, "A comprehensive survey on network anomaly detection," *Telecommun. Syst.*, vol. 70, no. 3, pp. 447–489, Mar. 2019.

[7] M. R. Belgaum, F. Ali, Z. Alansari, S. Musa, M. M. Alam, and M. S. Mazliham, "Artificial intelligence based reliable load balancing framework in software-defined networks," *Comput., Mater. Continua*, vol. 70, no. 1, pp. 251–266, 2022.

[8] M. Hamdan, E. Hassan, A. Abdelaziz, A. Elhigazi, B. Mohammed, S. Khan, A. V. Vasilakos, and M. N. Marsono, "A comprehensive survey of load balancing techniques in software-defined network," *J. Netw. Comput. Appl.*, vol. 174, Jan. 2021, Art. no. 102856.

[9] Z. Kabiri, B. Barekatain, and A. Avokh, "GOP-SDN: An enhanced load balancing method based on genetic and optimized particle swarm optimization algorithm in distributed SDNs," *Wireless Netw.*, vol. 28, no. 6, pp. 2533–2552, Aug. 2022.

[10] R. P. Dhanya and V. S. Anitha, "Implementation and performance evaluation of load balanced routing in SDN based fat tree data center," in *Proc. 6th Int. Conf. Inf. Syst. Comput. Netw. (ISCON)*, Mar. 2023, pp. 1–6.

[11] X. Diao, H. Gu, X. Yu, L. Qin, and C. Luo, "Flex: A flowlet-level load balancing based on load-adaptive timeout in DCN," *Future Gener. Comput. Syst.*, vol. 130, pp. 219–230, May 2022.

[12] S. Abdollahi, A. Deldari, H. Asadi, A. Montazerolghaem, and S. M. Mazinani, "Flow-aware forwarding in SDN datacenters using a knapsack-PSO-based solution," *IEEE Trans. Netw. Service Manage.*, vol. 18, no. 3, pp. 2902–2914, Sep. 2021.

[13] Q.-V. Pham, D. C. Nguyen, S. Mirjalili, D. T. Hoang, D. N. Nguyen, P. N. Pathirana, and W.-J. Hwang, "Swarm intelligence for next-generation networks: Recent advances and applications," *J. Netw. Comput. Appl.*, vol. 191, Oct. 2021, Art. no. 103141.

[14] W. Wong and C. I. Ming, "A review on metaheuristic algorithms: Recent trends, benchmarking and applications," in *Proc. 7th Int. Conf. Smart Comput. Commun. (ICSCC)*, Jun. 2019, pp. 1–5.

[15] F. Valdez, "A review of optimization swarm intelligence-inspired algorithms with type-2 fuzzy logic parameter adaptation," *Soft Comput.*, vol. 24, no. 1, pp. 215–226, Jan. 2020.

[16] R. A. Ammal and S. S. V. Chandra, "Bio-inspired algorithms for software defined network controllers," in *Proc. Int. CET Conf. Control, Commun., Comput. (IC)*, Jul. 2018, pp. 306–310.

[17] X. Wang, X. Li, and V. C. M. Leung, "Artificial intelligence-based techniques for emerging heterogeneous network: State of the arts, opportunities, and challenges," *IEEE Access*, vol. 3, pp. 1379–1391, 2015.

[18] A. H. Alhilali and A. Montazerolghaem, "Artificial intelligence based load balancing in SDN: A comprehensive survey," *Internet Things*, vol. 22, Jul. 2023, Art. no. 100814.

[19] K. Yu, L.-Z. Tan, X.-J. Wu, and W. Su, "Sonum: Software-defined optimal network utilization mechanism for long flow scheduling in data center networks," in *Proc. Int. Conf. Netw. Netw. Appl. (NaNA)*, Oct. 2019, pp. 149–154.

[20] S. Nabi, M. Ahmad, M. Ibrahim, and H. Hamam, "AdPSO: Adaptive PSO-based task scheduling approach for cloud computing," *Sensors*, vol. 22, no. 3, p. 920, Jan. 2022.

[21] F. Pasandideh, T. D. E. Silva, A. A. S. d. Silva, and E. P. de Freitas, "Topology management for flying ad hoc networks based on particle swarm optimization and software-defined networking," *Wireless Netw.*, vol. 28, pp. 257–272, Mar. 2022.

[22] M. S. Raghavendra, P. Chawla, and S. S. Gill, "DEEDSP: Deadline-aware and energy-efficient dynamic service placement in integrated Internet of Things and fog computing environments," *Trans. Emerg. Telecommun. Technol.*, vol. 32, no. 12, Dec. 2021, Art. no. e4368.

[23] S. Song, "Software defined network load balancing based on Im-ACO," in *Proc. IEEE Int. Conf. Control, Electron. Comput. Technol. (ICCECT)*, Apr. 2023, pp. 276–279.

[24] K. Sridevi and M. A. Saifulla, "LBABC: Distributed controller load balancing using artificial bee colony optimization in an SDN," *Peer Peer Netw. Appl.*, vol. 16, pp. 947–957, Feb. 2023.

[25] T. E. Ali, A. H. Morad, and M. A. Abdala, "Traffic management inside software-defined data centre networking," *Bull. Electr. Eng. Informat.*, vol. 9, no. 5, pp. 2045–2054, Oct. 2020.

[26] B. Nagasri and V. Anbarasu, "An optimization analysis on ant colony and particle swarm for load balancing techniques and policies with software defined networks (SDN)," in *Proc. Int. Conf. Innov. Comput., Intell. Commun. Smart Electr. Syst. (ICSES)*, Jul. 2022, pp. 1–6.

[27] CISCO. (2022). *IP Switching Cisco Express Forwarding Configuration Guide, Cisco IOS XE Release 3S*. Accessed: Jan. 20, 2022. [Online]. Available: https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/ipswitch_cef/configuration/xe-3s/isw-cef-xe-3s-book/isw-cef-load-balancing.html#GUID-5B419B21-DF72-4F5A-806D-EDB4FC0081CF

[28] F. Gul, W. Rahiman, S. S. N. Alhady, A. Ali, I. Mir, and A. Jalil, "Meta-heuristic approach for solving multi-objective path planning for autonomous guided robot using PSO–GWO optimization algorithm with evolutionary programming," *J. Ambient Intell. Humanized Comput.*, vol. 12, no. 7, pp. 7873–7890, Jul. 2021.

[29] W. Deng, J. Xu, and H. Zhao, "An improved ant colony optimization algorithm based on hybrid strategies for scheduling problem," *IEEE Access*, vol. 7, pp. 20281–20292, 2019.

[30] S. P. M. Ziyath and S. Senthilkumar, "RETRACTED ARTICLE: MHO: Meta heuristic optimization applied task scheduling with load balancing technique for cloud infrastructure services," *J. Ambient Intell. Humanized Comput.*, vol. 12, no. 6, pp. 6629–6638, Jun. 2021.

[31] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. Int. Conf. Neural Netw.*, Nov. 1995, pp. 1942–1948.

[32] M. Alotaibi and A. Nayak, "Linking handover delay to load balancing in SDN-based heterogeneous networks," *Comput. Commun.*, vol. 173, pp. 170–182, May 2021.

[33] V. Huang, G. Chen, P. Zhang, H. Li, C. Hu, T. Pan, and Q. Fu, "A scalable approach to SDN control plane management: High utilization comes with low latency," *IEEE Trans. Netw. Service Manage.*, vol. 17, no. 2, pp. 682–695, Jun. 2020.

[34] O. Romanov, I. Saychenko, A. Marinov, and S. Skolets, "Research of SDN network performance parameters using mininet network emulator," *Inf. Telecommun. Sci.*, no. 1, pp. 24–32, Jun. 2021.

[35] M. Hasan, H. Dahshan, E. Abdelwanees, and A. Elmoghazy, "SDN mininet emulator benchmarking and result analysis," in *Proc. 2nd Novel Intell. Lead. Emerg. Sci. Conf. (NILES)*, Oct. 2020, pp. 355–360.

[36] J. Chen, Y. Wang, X. Huang, X. Xie, H. Zhang, and X. Lu, "ALBLP: Adaptive load-balancing architecture based on link-state prediction in software-defined networking," *Wireless Commun. Mobile Comput.*, vol. 2022, pp. 1–16, Feb. 2022.

[37] T. Ethilu, A. Sathappan, and P. Rodrigues, "An efficient switch migration scheme for load balancing in software defined networking," *Int. J. Electr. Comput. Eng. Syst.*, vol. 14, no. 4, pp. 443–456, Apr. 2023.

MOHAMMAD RIYAZ BELGAUM received the master's degree with a specialization in computer applications, in 2001, the M.Eng. degree with a specialization in computer science and engineering, in 2006, and the Ph.D. degree in information technology from Universiti Kuala Lumpur, Malaysia, in 2022. He is an Oracle Database 11g Administrator Certified Professional and an Oracle Database 11g Program with PL/SQL Certified Associate.

He is currently an Associate Professor with the Department of Computer Science and Engineering, G. Pullaiah College of Engineering and Technology, Kurnool. He is also a Postdoctoral Researcher with Multimedia University, Malaysia, doing research on the application of blockchain for secure communication in WSN. He has authored more than 40 research papers published in journals and peer-reviewed conferences. His research interest include cloud computing, software defined networking, mobile ad hoc networking, wireless sensor networks, the Internet of Things, and bytecode analysis. He has vast teaching experience across various countries, with much focus on research.

SHAHRULNIZA MUSA received the Postgraduate Diploma degree in integrated research study and the Ph.D. degree in communication network security from the Faculty of Electrical and Electronic Engineering, Loughborough University, U.K., in 2005 and 2008, respectively. He is currently a Professor with Universiti Kuala Lumpur (UniKL). Apart from teaching and postgraduate supervision, he is also active in software project consultation and development in business applications. His research interests include cybersecurity, the IoT security, big data analytics, AI, and SDN.

FUEAD ALI received the B.Sc. degree in electrical and electronics engineering from Purdue University, the M.S. degree in computer science from Universiti Pertahanan Nasional Malaysia, and the P.Tech. degree.

He has been an academician with Universiti Kuala Lumpur-Malaysian Institute of Information Technology (MIIT), since 2009. Besides being an academician, he is also a Coordinator with the Corporate Management Office, UniKL Chancellery; mainly being tasked to monitor and manage the implementation of the high-profile projects which have been awarded to UniKL. Prior to joining UniKL, he was a Researcher with MIMOS Berhad and an Engineer with Intel Technologies Sdn Bhd and Western Digital Sdn Bhd. Until now, he has been involved in various research projects, locally and internationally and publishing research and technical papers in various conferences and journals. He is a member of MIET.

MUHAMMAD MANSOOR ALAM received the M.S. degree in system engineering, the M.Sc. degree in computer science, the Ph.D. degree in computer engineering, and the Ph.D. degree in electrical and electronics engineering in France, U.K., and Malaysia, respectively. He was the Associate Dean of CCSIS and the HoD of the Mathematics, Statistics, and Computer Science Departments, IoBM, Pakistan. He is recently associated with Riphah International University, Islamabad. He is a Professor of computer science. He has honor to work as an Online Laureate (Facilitator) for MSIS Program run by Colorado State University, USA, and Saudi Electronic University, Saudi Arabia. He also established research collaboration with Universiti Kuala Lumpur (UniKL) and Universiti Malaysia Pahang (UMP). He is currently an Adjunct Professor with UniKL and supervising 12 Ph.D. students. He has done Postdoctoral research on ‘‘machine learning approaches for efficient prediction and decision making’’ in Malaysia. He is enjoying 20 years of research and teaching experience in Canada, England, France, Malaysia, Saudi Arabia, and Bahrain, and authored more than 150 research articles which are published in well reputed journals of high impact factor, Springer Link book chapters, Scopus indexed journals and IEEE conferences. He received the Très Honorable (with distinction) From Universite de LaRochelle due to his research impact during his PhD.

ZAINAB ALANSARI (Member, IEEE) received the B.S. degree in computer science (summa cum laude) in 2007, the M.S. degree (Hons.) in computer system and networking from the AMA International University, Bahrain, and the Ph.D. degree in computer science from the University of Malaya (UM) in Malaysia. Since 2007, she has been a Lecturer and an Instructor of computer science in several institutions internationally. She is currently a Senior Lecturer with the University of Technology and Applied Sciences, Oman. She has published more than 40 ISI and Scopus-indexed articles and contributed to numerous conferences and seminars internationally as a presenter, a keynote speaker, and the session chair. Her research interests include the Internet of Things, wireless sensor networks, big data, and cloud computing. She received the Best Thesis Award for the M.S. degree from the AMA International University, in 2011.

SAFEEULLAH SOOMRO (Member, IEEE) received the B.Sc. (Hons.) and M.Sc. degrees in computer science from the University of Sindh, Jamshoro, Pakistan, and the Ph.D. degree in computer science from the Graz University of Technology, Austria. He has accomplished the Postdoctoral research in Europe. He has several years of teaching and research experience in national and international universities. Furthermore, he is a Senior Fellow with the Higher Education Academy, U.K., that ensure to teach and carry research in undergraduate and graduate universities around the world. He has written many research papers in conference proceedings, workshops, books, and journals. His research interests include the IoT, big data analysis, model based diagnosis and reasoning, software debugging and verification, formal methods, and software engineering. He is currently working on different projects on the IoT, program analysis, and model-based debugging. He is a member of ACM, IENG, IAER, IEEE Computer Society, and ISOC. He was elected as the Vice Chair of the NoVA IEEE Section, VA, USA. He is an editor of several books, journals, and IEEE and Springer conference proceedings in the field of computer science, software engineering, and information technology.

MAZLIHAM MOHD SU’UD received the master’s degree in electrical and electronics engineering from the University of Montpellier, in 1993, and the Ph.D. degree in computer engineering from Université de La Rochelle, in 2007. From 2013 to 2020, he was the President/CEO of Universiti Kuala Lumpur, Malaysia. Since 2020, he has been the President/CEO of Multimedia University, Malaysia. He has vast experience in publishing articles in high-quality international scientific journals and conference proceedings. He has numerous years of experience in the industrial and academic fields.