[1]Abdullah Ali Salman

Dr. Nasser Mozayani

The ACO Algorithm to Solve the Issue for a Traffic Engineering in the SDN

Abstract: - A While software-defined networking (SDN) offers flexibility and excellent programmability, it faces challenges in optimizing traffic engineering to adapt to changing network conditions. Enhancing SDN functionality relies heavily on effective traffic allocation and resource efficiency. Therefore, the necessity for an efficient, scalable, and adaptive algorithm is crucial to enable intelligent flow control and network improvement. Despite numerous proposed solutions within this realm, the absence of a precise and efficient algorithm for managing traffic in SDN networks remains apparent due to the complexity of the problem. In this study, an enhanced ACO algorithm was implemented for traffic management in an SDN network. The findings from simulations conducted on various network topologies indicate that the ACO algorithm significantly enhances network traffic distribution by identifying more optimal paths, thereby promoting load balancing within the network.

Keywords: Traffic Engineering(TE), Software Defined Networks(SDN) ,Swarm Intelligence (SI), Ant Colony Optimization Algorithm (ACO)Load balancing (LB)

1. Introduction

Software Defined Networking (SDN) is considered as a promising approach in networking paradigm. It distinguishes between the network's control plane and the data forwarding plane. This strategy not only aids in the optimal exploitation of network resources, but it also decreases network administration complexity, lowers network operating costs, and fosters novel and evolutionary ideas.

It separates the network control logic from the underlying routers and switches, encourages centralized network control, and enables network operation programming. SDN will become the preferred platform for deploying numerous networks. Compared to conventional networks. OpenFlow serves as the prevalent communication protocol connecting the controller plane with the switch plane in SDN. SDNs enable centralized control, which provides a global view of the network and allows for dynamic reconfiguration to adapt to changing traffic patterns. Techniques such as adaptive robust traffic engineering and dynamic traffic engineering leverage this capability to optimize network performance and reduce reconfiguration frequency, ensuring stability and minimizing overhead [1, 2].

Traffic engineering (TE) in SDNs involves optimizing the management of network traffic to improve performance, reliability, and resource utilization. This is achieved by leveraging the decoupled control and data planes, which allow for centralized network management and dynamic reconfiguration. SDN traffic engineering challenges include flow management, fault tolerance, topology update, and traffic analysis, requiring novel solutions to optimize performance and manage traffic in software-defined networks. Traffic engineering can reduce connection failures and service degradation in networks.. [3]Therefore the effective traffic engineering in SDNs involves a combination of centralized control, dynamic reconfiguration, load balancing, advanced machine learning techniques, hybrid deployments, and robust multi-controller architectures. These approaches collectively optimize network performance, resource utilization, and resilience. Cluster-based routing and multi-controller architectures enhance the scalability and resilience of SDNs. These approaches reduce control traffic delay and improve fault tolerance, ensuring efficient network operation even during traffic spikes or controller

[1] [1] School of Computer Eng. Iran University of Science and Technology , Tehran, Iran

[2]Associate Professor, School of Computer Eng .Iran University of Science and Technology

Narmak, Tehran, Iran ,Website: http://webpage.iust.ac.ir/mozayani/

Phone: (+98) 21-7322-5300, (+98) 21-7322-5366

Email : corresponding : mozayani@iust.ac.ir [*2] , a_abdullah95@yahoo.com [1]

Copyright©JES2024on-line:journal.esrgroups.org

failures [4, 5] Advanced techniques such as machine learning, reinforcement learning and Genetic Algorithm are being used to enhance traffic engineering in SDNs. These methods can predict traffic patterns and dynamically adjust routing decisions to optimize performance and reduce latency [6, 7]

Ant Colony Optimization (ACO) utilizes a probabilistic method to address the target problem, similar to GA, simulated annealing, and other algorithms that rely on heuristics. Inspired by the behavior of ants, the ACO algorithm involves ants initially wandering aimlessly before returning to the nest after depositing pheromones on paths while searching for food. Subsequent ants can then follow these pheromone trails instead of moving randomly. This strategy has been applied to tackle issues in computer networking, such as determining the optimal path within the network. Various systems based on the ACO algorithm, such as AntNet, AntHocNet, HopNet, and Stigmetry, can be utilized to solve the routing issue for the computer network.[8, 9] In this study, we implemented a Swarm Intelligence-based Traffic Engineering (SITE) algorithm which is essential for intelligent traffic management and network optimization.

2. RELATED WORK

Qi et al. [10] decided to better learn the traffic features and improve the routing performance, proposed a TE approach combining contrastive learning and reinforcement learning to optimize routing of network traffic in hybrid SDN. In this paper each agent trains an encoder that well represents the traffic features through contrastive learning and the traffic features are fed into the training of the actor neural network for learning the map between the traffic and routing policy through reinforcement learning. After receiving offline training, the agent installed on the SDN switch may quickly infer an appropriate traffic splitting policy that defines the traffic splitting ratio on the switch.. Extensive experiments on three different network topologies showed that their proposed algorithm provides significant improvements.

In [10] a load-balancing technique is presented which averages the load across SDN controllers, thereby facilitating effective load distribution. The performance of the proposed technique is evaluated based on the degree of load balancing, network response time, and migration cost. These metrics provide insights into the method's efficacy in maintaining load stability and the overhead associated with the migration process. The proposed approach showed that the load imbalance degree of the proposed method decreased by an average of 53.3% and 43.5% compared to EASM and SMS respectively.

Authors in [11] have proposed the migration of the data plane components to balance the load between distributed SDN controllers. Different from most previous works which use reactive mechanisms, wethey proposed to preemptively balance the load in the SDN control plane to support network flows that require low latency communications. First, they forecast the load of SDN controllers to prevent load imbalances and schedule data plane migrations in advance. Second, they optimized the migration operations to achieve better load balancing under delay constraints by constructing two prediction models based on Auto Regressive Integrated Moving Average (ARIMA) and Long Short-Term Memory (LSTM) techniques for forecasting SDN controller demand. Their results show that, in long-term predictions, the accuracy of LSTM model outperforms that of ARIMA by 55% in terms of prediction errors. The simulations reveal that the suggested approach works near optimally and outperforms previous benchmark algorithms from the literature.[12]

Guo et al. [13] proposed an approach that leverages Graph Neural Networks (GNNs) and multi-arm bandit algorithms to dynamically optimize traffic management policies based on real-time network traffic patterns. They used a GNN model to learn and predict network traffic patterns and a multi-arm bandit algorithm to optimize traffic management policies based on these predictions. They evaluated their proposed approach on three different datasets, including a simulated corporate network (KDD Cup 1999), a collection of network traffic traces (CAIDA), and a simulated network environment with both normal and malicious traffic (NSL-KDD). Their results demonstrated that the approach outperforms other state-of-the-art traffic management methods, achieving higher throughput, lower packet loss, and lower delay, while effectively detecting anomalous traffic patterns.

There exist various studies on TE of SDN [14] presented a scheme on how to use the genetic algorithm (GA) for solving the LB problem of SDN. Although it is an innovative method, the conventional GA is extremely time-consuming.. There are also studies adopting Ant Colony Optimization (ACO) for LB [15-18].

A machine learning-based meta-layer and heuristic algorithm layer comprise the traffic-engineering framework that was suggested. The meta-layer trains the heuristic algorithm layer in traffic engineering to identify the best route following training. The optimum path is then used for dynamic routing, which produces effective network operation. Additionally, QoE-Centric flow routing using ACO, [16] which was shown to be better than Shortest Path Routing (SPR). ACO was paired with job categorization for multi-controllers, which separates central controllers from sub-controllers.[15].

In [8] a traffic engineering framework consisting of a heuristic algorithm layer and a machine learning-based meta layer is proposed. For traffic engineering, the heuristic algorithm layer is trained by the meta layer to find the optimal path. Dynamic routing is then performed on the optimal path, resulting in efficient network operation. ACO is also used for QoE-centric flow routing, which was shown to be better than Shortest Path Routing (SPR).

ACO is combined with multi-controller job classification, where the job classification distinguishes the central controller from the sub-controllers. While the basic ACO algorithm produces good results, it requires a lot of computation before it can fully function. As with typical optimization problems, the solution may take a long time to converge or reach a local optimum. It is also not easy to deploy them on the network when the operating scenario changes. This paper classifies AI-based load balancing technologies and carefully evaluates these mechanisms from different perspectives, including the algorithms/methods used, the problems solved, and their advantages and disadvantages. Third, it summarizes the indicators used to evaluate the effectiveness of these strategies. Finally, identifying the tendencies and demanding situations of AI-based load balancing for destiny research. SDN architecture is organized into three principal planes based on the Open Networking Foundation (ONF). [19]

Although the basic ACO algorithm produces good results, it requires a large number of computations to reach its full potential. Similar to other common optimization issues, it could take a while for the solution to converge or reach the local optimum. Deploying to the network when the operation scenario changes is also difficult.

3. PROPOSED METHOD

The problem of this research is the use of ACO algorithm for traffic engineering in an SDN network. Traffic engineering in SDN network using ACO algorithm includes dynamic optimization of traffic routing to increase network efficiency. The problem of traffic engineering in this research is to find optimal paths in order to balance the load:

$$paths = \min_{V,E} f(Packest) \quad (1)$$

$paths$ represents the paths in the network. V and E represent the set of switches and network links, respectively. $f(Packest)$is a function that represents the amount of packets passing through network links. The function of the number of packets traveling over network links is represented by f in this context. In order to achieve load balancing, the process of uniformly distributing traffic load throughout the network aims to avoid congestion and guarantee the effective use of available resources. By mimicking the actions of ants searching for the best roads based on pheromone pathways and heuristic information, the ACO algorithm finds efficient paths. While maximizing throughput, these effective pathways also decrease latency. Additionally, these pathways result in a more stable network by preventing overloading of each particular link.

3.1 Flowchart OF Proposed Method

The suggested method's flowchart is displayed in the accompanying diagram. First, the network topology is defined using this flowchart. Defining the network topology includes identifying the amount of switches, links, and network configuration. Once the topology has been established, the switches are initialized and ant farms are randomly assigned to them. The routing operation is then carried out by determining the probability of each path based on the pheromone material. Until the optimal pathways achieve the ultimate requirements, these procedures are repeated. Ultimately, this algorithm's output will balance the load and improve network efficiency.

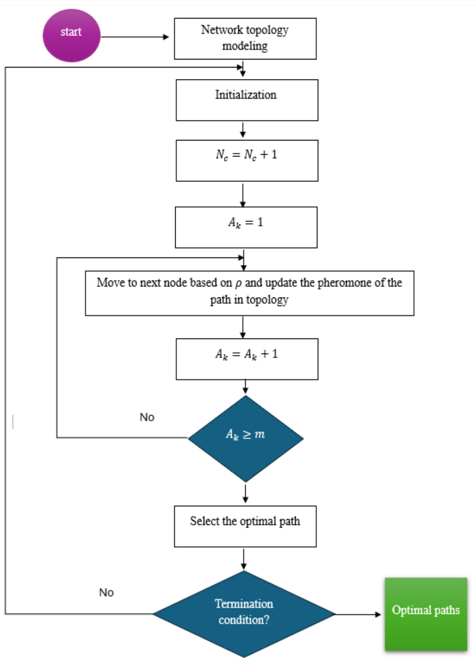

Figure Error! No text of specified style in document.**1**. Flowchart of the proposed method

3.1.1 Network Topology Modeling

At the beginning of the proposed method, the network topology is modeled. A two-dimensional graph is used to model the network. Graph G is a set of nodes (switches) and edges (links). Here, graph G can be modeled as equation 2:

$$G_{SDN} = (V, E) \quad (2)$$

In graph-based topology, links are communication paths between switches. Each link has three characteristics: C_{ij} , L_{ij} , D_{ij} (Figure 3-2). Therefore, the 5 parameters of Table 1 represent the main characteristics of the graph that should be considered in modeling.

Table 1. main parameters in SDN topology modeling

Variable	Description
V	Set of nodes (switches)
E	Set of edges (links)
C_{ij}	Capacity of link
L_{ij}	Current load in link
D_{ij}	Delay of link

In the modeled graph, the number of V and E is very important in the problem of path finding and load balancing. It is assumed that the studied network has m switches and k links It depends on (Capacity of link, Current load in link, Delay of link).

$$V = \{V_1, V_2, \dots, V_n\} \quad (3)$$

$$E = \{E_1, E_2, \dots, E_k\} \quad (4)$$

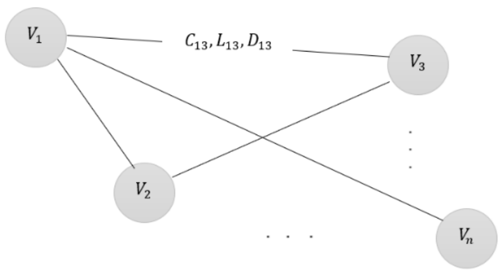

Figure 2. A simple example topology with C_{13}, L_{13}, D_{13}

3.1.2 Initial Settings

Initial settings are created after the network topology has been defined and modeled. They serve to define the network static variables, such as the number of m-k, the minimum acceptable delay between two switches, the initial parameters of the ACO optimization algorithm, and other parameters required to initiate path finding and load balancing. This step in the simulation process is crucial because a network whose parameters do not correspond to real-world conditions cannot yield trustworthy results

3.1.3 Definition of fitness function

The fitness function is defined as shown below. The purpose of ACO optimization is to locate efficient pathways and equally distribute traffic across the network. To balance the network load, the ACO algorithm should find the optimum pathways based on the following relationship.

$$F = \max\left(\frac{L_{ij}}{C_{ij}}\right) + \delta \sum_{(i,j)\epsilon E} D_{ij}.x_{ij} \quad (5)$$

L_{ij}	Current load in link
C_{ij}	Capacity of link
D_{ij}	Delay of link

In this regard, δ determines the importance of load uniformity. Also, in this relation, x_{ij} is a binary variable. This variable is equal to one if the link (i,j) is used in the selected path; otherwise it is zero.

$$x_{ij} = \begin{cases} 1 & if\ (i,j) \in pathsE \\ 0 & if\ (i,j) \nexists\ pathsE \end{cases} \quad (6)$$

$$\in \backslash belong\ to$$

$$\nexists \backslash no\ belong\ to$$

3.2. ACO optimization algorithm in SDN traffic engineering

The ACO algorithm is derived from the behavior of ants, where they leave pheromone trails from the nest to the food source, which is then followed by other ants to obtain the food.

The ACO algorithm is used to traffic engineering in two parts. In the first stage, ants find new trails and collect information about current ones. This process is known as a forward update. If one of the ants successfully reaches the destination node, some of them will return to the source node via the already traveled path in the second phase. This round trip updates the nodes' routing tables.. This step is called a backward update. With SDN, these updates are eliminated because the controller has an overview of the entire network[20].

To demonstrate the concept of ACO in the proposed scheme, a practical SDN topology has been used as shown in the Figure **Error! No text of specified style in document.**. OpenFlow plays a fundamental role in SDN topology by providing a standard protocol, which handles direct communication between the SDN controller and network devices such as switches and routers. This protocol facilitates the separation of the control plane from the data plane and makes the controller centrally manage the network behavior by dynamically configuring the flow tables in OpenFlow switches. This feature enables the SDN controller to optimize traffic flows, implement dynamic routing policies, and thereby perform real-time traffic engineering. Obviously, traffic management has resulted in efficient load balancing throughout the network, which ultimately increases the overall performance and flexibility of the network.

In order to illustrate the concept of ACO employed in the proposed scheme, the topology shown in Figure **Error! No text of specified style in document.** is used. Each ant in the ant-world may be considered as a network packet that begins randomly at any node in the topology.

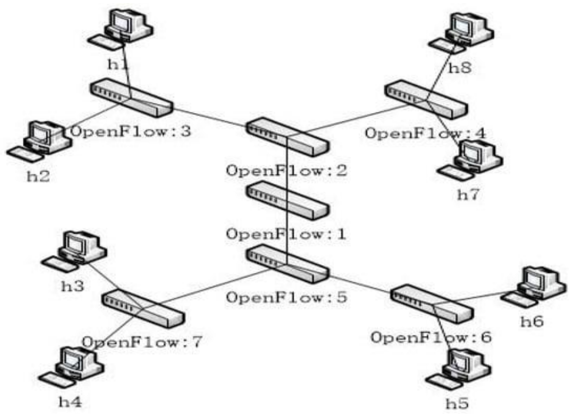

Figure Error! No text of specified style in document.. Topology based on Open flow[15]

As previously stated, the number of switches in the topology is considered to be equal to n. To implement the ACO algorithm for traffic engineering on links and switches, a clear definition of ants is required. In this study's problem, each ant is treated as a packet delivered over a network. To do this, each ant's movement begins at random in each network node. This indicates that the ants have a random movement location, and the commencement of this movement comes from one of the network switches. Now, if $b_i(t)$ stands for the number of ants in the ith switch at time t, then the following relationship between m and n and $b_i(t)$ is established.

$$m = b_1(t) + b_2(t) + \cdots + b_n(t) \qquad (7)$$

In this regard, m is the total number of ants, which can be written as follows. This relationship shows that the total number of ants is obtained from the total number of ants in the switches:

$$m = \sum_{i=1}^{n} b_i(t) \qquad (8)$$

It is vital to note that an ant does not visit the same switch repeatedly. This condition prevents loops and lowers efficiency. To provide these circumstances, temporary memory is required. As a result, each ant maintains track of which switches it has visited in a taboo table. This method allows for the discovery of more optimum pathways in the network, resulting in more balanced traffic distribution. To model the state transfer probability between two switches based on the remaining pheromone in each path (probability of ant k moving from link i to j), the following equation is used:

$$P(A_k)_{ij} = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha}.\mu_{ij}^{\beta}}{\sum_{a_k}[\tau_{ij}(t)]^{\alpha}.\mu_{ij}^{\beta}}, & j \in a_k \\ 0 & o.w \end{cases} \quad (9)$$

$$A_k = 1,2,\dots,m$$

In this structure , k is the ant number A_k and $P(A_k)_{ij}$ models the probability of Ak for Vj to visit Vi. Also, a_k shows the available switches to select when Ak is in Vi. Also, α represents the weight of the remaining pheromone. β also represents the path distance corresponding to the remaining pheromone. In this relation, τ_ij (t) also models the entire pheromone that is located in the path between Vi and Vj. The amount of pheromone in this path at the beginning (initial value) is equal to θ.

$$\tau_{ij}(0) = \theta \qquad (10)$$

Pheromone is one of the key variables in the ACO algorithm for traffic engineering. These compounds are disseminated by ants throughout the pathways. The amount of pheromone on a connection determines how appealing it is to ants. As a result, the higher the pheromone level, the more probable the link will be picked. The figure below shows how the Ak ant chooses the lower path based on more pheromone (higher probability).

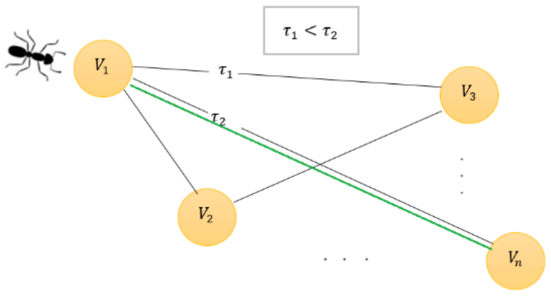

Figure Error! No text of specified style in document.3 (Representation of signal stages of two-junction network)Paths with $\tau_1 < \tau_2$

This relationship is very important in pathfinding and traffic engineering because ants based on the probability calculated by this relationship; They choose the next switch to visit, in other words, this relationship is the decision to choose the next switch. In the above relation, μ_{ij} is also calculated with the following relation. This variable indicates the intrinsic-exploratory quality of the link. In this regard, dis_{ij} represents the distance between two switches i and j.

$$\mu_{ij} = \frac{1}{dis_{ij}} \qquad (11)$$

By placing dis_{ij} instead of μ_{ij} in the visit probability relationship, the following relationship is obtained:

$$P(A_k)_{ij} = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha}.\left(\frac{1}{dis_{ij}}\right)^{\beta}}{\sum_{a_k}[\tau_{ij}(t)]^{\alpha}.\left(\frac{1}{dis_{ij}}\right)^{\beta}}, & j \in a_k \\ 0 & o.w \end{cases} \quad (12)$$

This formula shows that the smaller the distance between i and j, the more likely the ant will visit node i. In this regard, there are two important parameters. These two parameters are very effective on the main relationship. The small value of α makes the probability of the next switch completely depend on dis_{ij}, which is not a desirable result. This problem causes local optima. On the other hand, if β is small, the next switch is chosen randomly, which results in an inaccurate answer to the problem.

Table 2. Analysis of two parameters α and β

parameter	Effective parameter on $P(A_k)_{ij}$	The problem of being small
α	$\tau_{ij}(t)$	local optimal solution
β	μ_{ij}	not guarantee finding any good solution

for solving this problem, The pheromone volatilization relationship is used. The relation of pheromone volatilization from the link is equal to:

$$\tau_{ij}(t+n) = (1-\rho) * \tau_{ij}(t) + \Delta\tau_{ij}(t) \quad (13)$$

In the above relation, ρ represents the volatilization percentage of pheromone on link ij. $\Delta\tau_{ij}(t)$ also shows the increase or decrease of pheromone along the path based on the ant's movement. The existence of this relationship is very essential in finding optimal paths. This relationship (pheromone volatilization) helps to control and reduce the amount of pheromone on the paths over time and thus prevents the accumulation of pheromone on the paths that were temporarily attractive but may no longer be optimal. In relation $\Delta\tau_{ij}(t)$ is obtained using the following formula:

$$\Delta\tau_{ij}(t) = \Delta\tau_{ij}^{1}(t) + \Delta\tau_{ij}^{2}(t) + \cdots + \Delta\tau_{ij}^{m}(t) = \sum_{k=1}^{m} \Delta\tau_{ij}^{k}(t) \qquad (14)$$

The initial value of $\Delta\tau_{ij}(t)$ is zero. This value indicates the initial state of the path[21].

$$\Delta\tau_{ij}(t) = 0$$

In this regard, $\Delta\tau_{ij}(t)$ has two modes in modeling the rate of pheromone volatilization. The first mode is positive feedback. Positive feedback means reinforcement of optimized paths by ants. When an ant travels a path and reaches its destination, the amount of pheromone on that path increases (positive). This increase in pheromone increases the probability of choosing this path by the next ants. Therefore, this feature is modeled in the form of the following relationship.

$$\Delta\tau_{ij}(t) = \begin{cases} \frac{Q}{L_k} & if\ A_k\ moves\ from\ Vi\ to\ V_j \\ 0 & o.w \end{cases} \quad (15)$$

The second mode is negative feedback. This is the opposite of the behavior of the ant in the state of crossing a path.

In the above relations, there are four main parameters which are defined as follows.

Table 3. Definition of main variables in ACO relations for SITE

Definition	Variable
Number of Ants	m
Pheromone Importance	α

Heuristic Importance	β
Evaporation Rate	ρ

The fundamental purpose of the ACO algorithm is to identify optimum pathways using the concept of ants traveling around a network. Each ant that moves from one switch to another makes a record (pheromone). These records are represented using a probabilistic connection. Routing activities include estimating the likelihood of each path depending on the pheromone ingredient. These stages are continued until the optimum pathways achieve the desired criteria. Finally, the result of this method will balance the load and improve the network performance. This approach is known as network traffic engineering.

3.3. The Permanent Structured Cooperation (PESCO) code

The summary of relationships and algorithm presented in this research is summarized in the following 10 steps.

Pesco code

1. The network graph is modeled.

$$G = (V, E)$$

2. It is assumed that there are n switches in the network.

$$V = \{V_1, V_2, \dots, V_n\}$$

3. It is assumed that there are k links in the network.

$$E = \{E_1, E_2, \dots, E_k\}$$

4. The number of ants (variables of ACO algorithm) is set.

$$m = set\ as\ num$$

5. The constants of the ACO algorithm are set.

$$(\alpha - \beta - m - \rho)$$

6. Fitness function (balance in load) is defined.

$$F = max\left(\frac{L_{ij}}{C_{ij}}\right) + \delta \sum_{(i,j)\epsilon E} D_{ij}.x_{ij}$$

7. Ants find the most efficient paths using the following probabilistic relations.

$$m = \sum_{i=1}^{n} b_i(t)$$

$$P(A_k)_{ij} = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha}.\mu_{ij}^{\beta}}{\sum_{a_k}[\tau_{ij}(t)]^{\alpha}.\mu_{ij}^{\beta}}, & j \in a_k \\ 0 & o.w \end{cases}$$

$$\tau_{ij}(0) = \theta$$

$$\mu_{ij} = \frac{1}{dis_{ij}}$$

$$\tau_{ij}(t+n) = (1-\rho) * \tau_{ij}(t) + \Delta\tau_{ij}(t)$$

$$\Delta\tau_{ij}(t) = \Delta\tau_{ij}^{1}(t) + \Delta\tau_{ij}^{2}(t) + \cdots + \Delta\tau_{ij}^{m}(t) = \sum_{k=1}^{m} \Delta\tau_{ij}^{k}(t)$$

$$\Delta\tau_{ij}(t) = \begin{cases} \dfrac{Q}{L_k} & if\ A_k\ moves\ from\ Vi\ to\ V_j \\ 0 & o.w \end{cases}$$

8. Has the minimum error condition been met?
If the answer is yes, go to 9, otherwise, go to 7
9. Efficient paths are introduced in the output of the algorithm.
10. Go to 1

3.4 SIMULATION

The primary objective of this study was to build and evaluate a traffic engineering strategy based on swarm intelligence (SITE) for SDN utilizing the AGIS network topology and preset parameters. Specific targets include:

Development of a Python-based simulation environment to simulate an SDN network based on AGIS topology.

Implementation of swarm intelligence algorithms, such as Ant Colony Optimization (ACO) for dynamic allocation of flow to path and traffic optimization and ultimately increasing network capability

Performance evaluation of SITE compared to traditional SDN TE algorithms.

In this study, the Mininet-WiFi library and other SDN simulation tools are used to generate a simulated SDN environment based on the AGIS network topology, which is then tested against various traffic scenarios to assess the SITE implementation's performance. Mininet is a frequently used network simulator in this field of study. This simulator creates a network of virtual hosts, switches, controllers and links (network graph)[22]. It hosts standard Linux network software and switches also use OpenFlow for flexible routing with SDN. According to these features, Mininet has been used for simulation so that the performance of the proposed method can be compared[23].

3.4.1 Simulation constants

The selection of ACO algorithm simulation constants has a significant impact on the final output. Thus, adjusting these factors may result in improved outcomes. One of the advances of this research is adjusting these settings to achieve better outcomes. The table below shows the size and definition of the default constants in the ACO algorithm for traffic engineering[24].

Table 4. Size and definition of default constants in ACO algorithm

Definition	Value	Variable
Number of Ants	8	m
Number of Iterations	100	max_iter
Pheromone Importance	1	α
Heuristic Importance	2	β
Evaporation Rate	0.4	ρ

The table below also illustrates the size and definition of the default constants in the customized ACO method for the traffic engineering problem in SDN. One of our advances was to improve the ACO algorithm for the specific usage of swarm intelligence-based traffic engineering (SITE) in SDN. These parameters are obtained based on trial and error. These changes are made to avoid creating local optima.

Table Error! No text of specified style in document.. Size and definition of improved constants in ACO algorithm

Definition	Value	Variable
Number of Ants	8	m
Number of Iterations	100	max_iter
Pheromone Importance	2.5	α
Heuristic Importance	3.1	β
Evaporation Rate	0.55	ρ

3.5 Implementation steps

The following steps have been taken to implement the proposed method.

Get data from Zip file: First, a Zip file called "topology-zoo.zip" is downloaded from Google Drive, which contains network data. This file is downloaded from Google Drive using PyDrive.Data Extraction: After downloading, the Zip file will be extracted to access the network data.Processing GML files: Network data is read from GML files. This process involves reading the GML files in the extracted directory and creating the corresponding networks using the NetworkX library.Network performance analysis: For each network, its performance is checked before the ACO algorithm and after the implementation of the ACO algorithm. The network performance includes the average and maximum load of nodes. Also, the ACO algorithm is implemented to optimize the traffic in the network, and the traffic before and after the optimization is displayed[25].Statistical analysis: Various statistics of networks are calculated, including average degree of nodes, coefficient of variance of degree of nodes, correlation between closeness centrality and degree of nodes, degree assemblage, neighborhood size of the largest node, 2-core size and pyramid fit.

Histogram analysis: The histogram of the average degree of nodes is drawn for the distribution of the average degree of nodes. Displaying results: The results of network analysis and related functions, including histogram and analytical statistics, are displayed. Network monitoring: A separate thread is run to monitor the network, which shows the amount of bytes sent and received continuously. Finally, the program uses another thread to monitor the network to continue the main process while monitoring.

4. Experimentation and Results

4.1 Simulation results

This section presents the final simulation results. This application produces graphs linked to the histogram of the average degree of nodes, as well as other graphs relevant to network research. Analytical statistics for each network are also displayed, including the average degree of nodes, the coefficient of variance of the degree of nodes, the correlation between the centrality of closeness and the degree of nodes, the degree assemblage, the size of the neighborhood of the largest node, the 2-core size, and the Pyramid fit. An example of various complicated networks is investigated in the file "topology-zoo.zip". This output provides the following information gathered during the program's execution: Information about the number of bytes delivered and received from the network over time:

Different values are displayed for bytes sent and received from the network. This information can be useful for monitoring network activity and its connections with other networks. Analytical statistics related to networks:

Various statistics have been calculated for each network, including the average degree of nodes, the coefficient of variance of the degree of nodes, the correlation between the centrality of closeness and the degree of nodes, degree assemblage, the neighborhood size of the largest node, 2-core size and pyramid fitting.

This information examines important characteristics of networks such as alignment, connectivity between nodes, and location of important nodes.

In general, the comparison has been made for four networks.

Comparison of two modes (without ACO and with ACO) for Cynet network

Comparison of two modes (without ACO and with ACO) for janetbackbone network

Comparison of two modes (without ACO and with ACO) for Amres network

Comparison of two modes (without ACO and with ACO) for AGIS network

In the table below, the specifications of the four grids (size V and E) are listed.

Table 6. characteristics of the four networks (size V and E) under study

NET	V	E
Cynet	10	20

janetbackbone	15	25
Amres	8	12
AGIS	12	18

The graphics in the figures below demonstrate the influence of the ant colony optimization algorithm (ACO) on various network topologies. The photos labelled "Traffic Before ACO" depict network traffic before the ACO algorithm is deployed. You can observe that the traffic distribution is frequently unequal, with some edges receiving much more traffic than others.

The photos labeled "Traffic After ACO" depict network traffic following the application of the ACO algorithm. The traffic distribution is more balanced, since the ACO algorithm finds more efficient pathways and distributes traffic equally over the network.

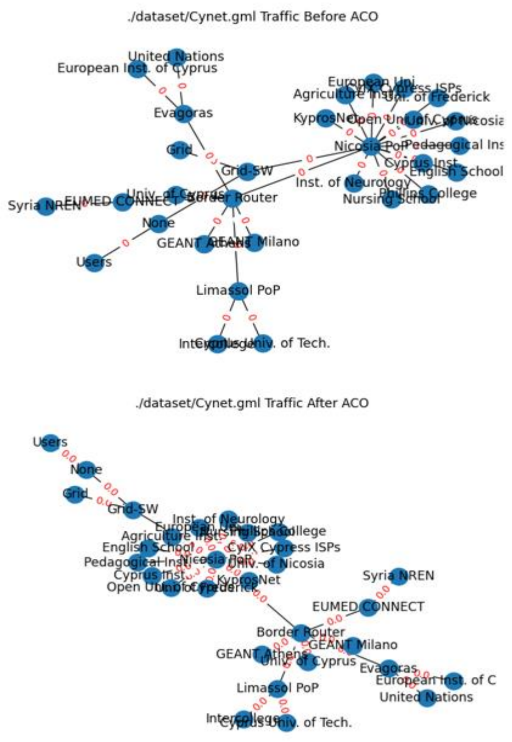

Figure 5. Comparison of two modes (without ACO and with ACO) for Cynet network

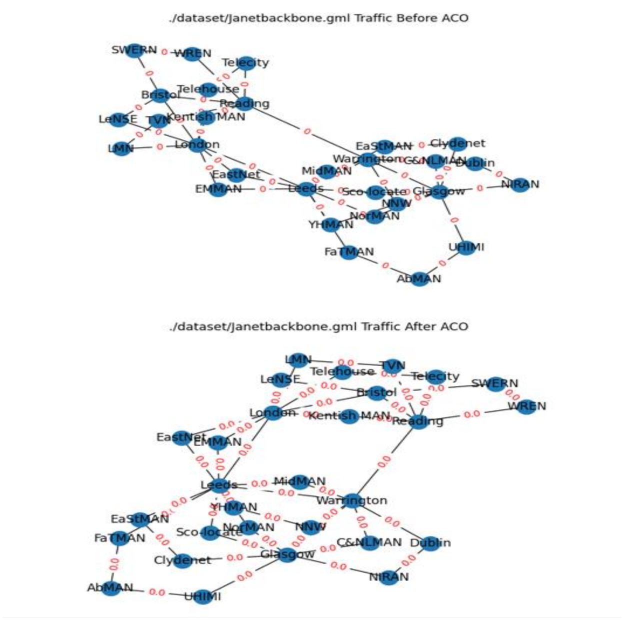

Figure 6. Comparison of two modes (without ACO and with ACO) for janetbackbone network

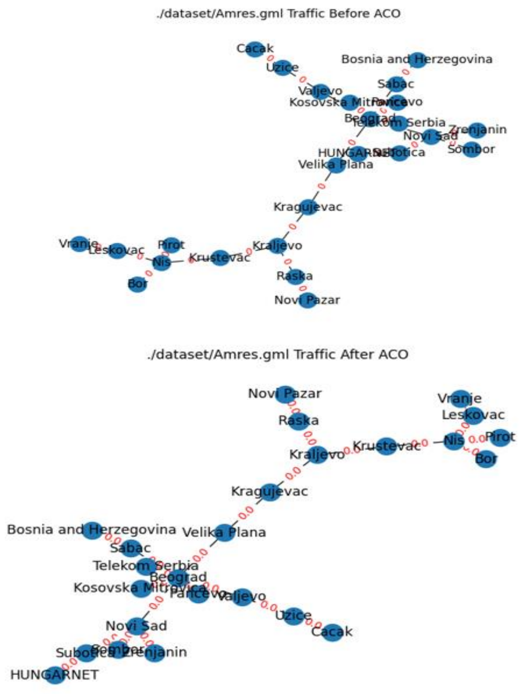

Figure 7. Comparison of two modes (without ACO and with ACO) for Amres network

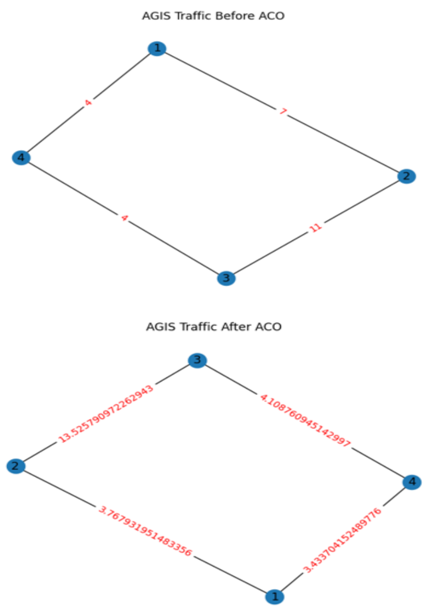

Figure 8. Comparison of two modes (without ACO and with ACO) for AGIS network

The results clearly show that the use of ACO in these networks has led to the improvement of traffic management, congestion reduction and path optimization[26] The most noteworthy aspect of the occurrence is the uniformity of traffic size across network pathways following the use of the ACO algorithm. This highlights the relevance of the suggested technique, particularly for bigger and more complicated networks like Janetbackbone. If ACO fails to employ the route optimizer, some paths may get saturated with traffic, causing network delays and inefficiencies. The graphic below shows a comparison of network size and CDF for several networks.

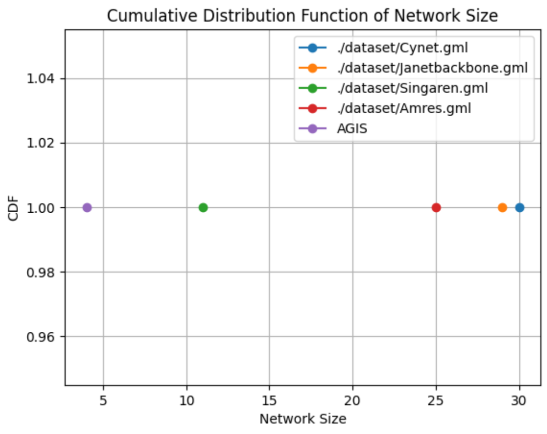

Figure 9. Comparison between Network size and CDF for different networks

The CDF plot shows the cumulative distribution of network sizes for different network topologies. This figure shows that most networks are about 10-30 nodes in size.

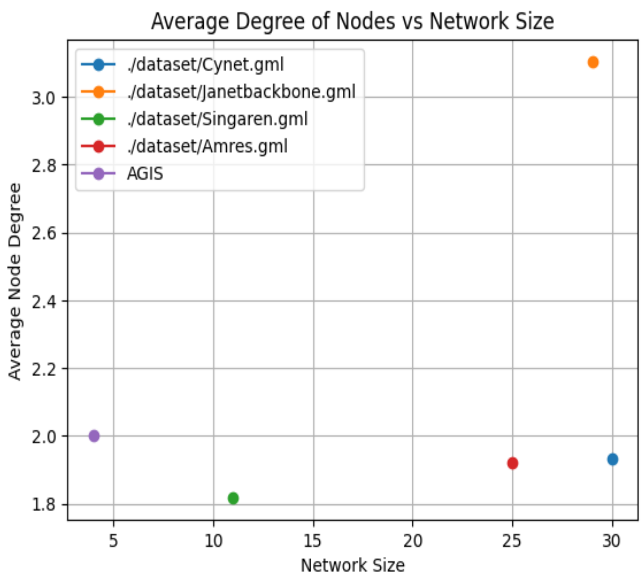

Figure 10. Comparison between Network size and Average node degree for different networks

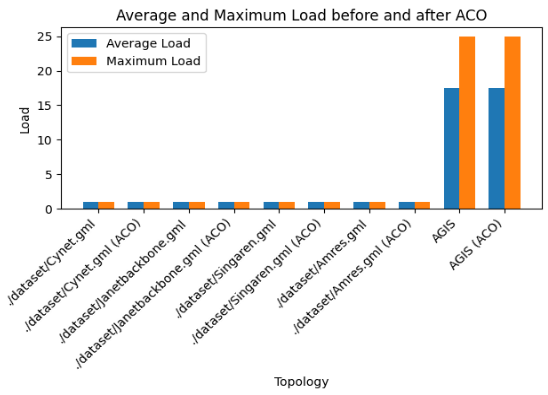

Figure 11. Comparison between the amount of load in two modes of ACO and without ACO for different networks

The ACO algorithm clearly improves network traffic distribution by identifying more efficient pathways, resulting in a more balanced load at network edges. The exhibited pictures demonstrate the usefulness of the ACO algorithm in improving network traffic allocation. The analysis of the network characteristics shows that the ACO algorithm is especially useful for medium-sized and dense networks. The results show the importance of using optimization algorithms for efficient network management and resource allocation[27].

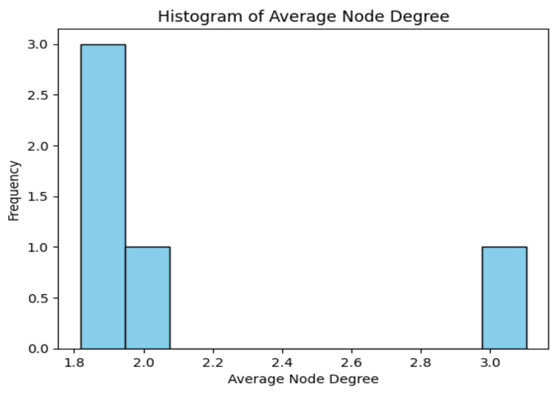

Figure 12. Average node degree histogram

The histogram shows the average frequency of different node degrees across the networks. This bar graph shows that most networks have an average node degree between 1.8 and 2.0, indicating a relatively sparse network structure. In summary, CDF plots and histograms show that the analyzed networks are generally of medium size and relatively sparse in terms of connections.

Authors should describe the results and how they might be interpreted in the context of prior investigations and the working hypothesis. The findings and their implications should be discussed in the broadest context possible. Future research directions might also be mentioned.

4.1.2 Discussion and comparison

In the table below, a comparison is made between Set1 and Set2. set1 is a set of parameters that used the ACO algorithm by default in the traffic engineering process. set2 also represents the set of parameters set for ACO (as custom). For better comparison in both tables the mode without ACO is also included. As it is known, the results are better in set2 mode.

Table 7. TE and SITE comparison for set1

$Set1$	Success rate
TE (no ACO)	89.5
SITE (with ACO)	91

Table 8. TE and SITE comparison for set2

$set2$	Success rate
TE (no ACO)	89.5
SITE (with ACO)	96.3

The results show that although the basic ACO algorithm provides good results, a lot of calculations are needed before it is fully operational. As with conventional optimization problems, the TE solution may take a long time to converge or lead to a local optimum. Also, it is not easy to deploy in the network when the operation scenario changes. This problem was solved to some extent in our research.G-ACO is a new solution to LB from SDN that blends GA and ACO. The present system, based on the ACO algorithm, employs a positive feedback mechanism to update the Path information of streams as they are transmitted. However, this may lead to a local optimal solution and inappropriate route selection. Also, the proposed method in [15]has a computational burden. In this research, we have shown that by presenting an improved plan in ACO, we can perform LB operations with good accuracy despite the absence of processing load. In the chart below, a complete comparison between the three modes is made

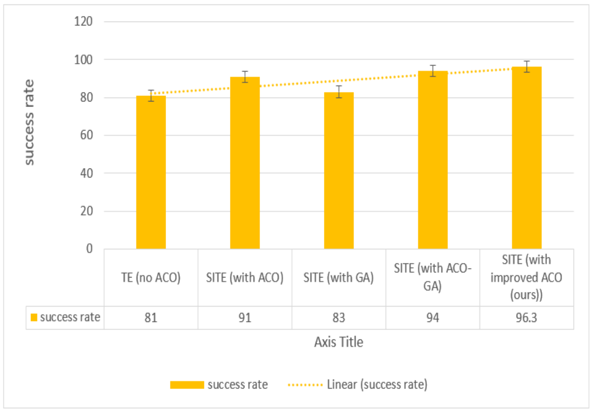

Figure 13. Comparison between different TE methods

5. Conclusions and Future Work

Efficient traffic distribution, swarm avoidance, and resource utilization are critical to enhance SDN performance. In this context, the implementation of swarm intelligence-based traffic engineering (SITE) algorithms is essential for intelligent flow management and network optimization. Despite presenting various researches in this field, the lack of an accurate and fast algorithm for traffic engineering in SDN network is still felt. For this purpose, in this research, the improved ACO algorithm was used for traffic engineering in an SDN network. The simulation results on different topologies show that the ACO algorithm obviously improves the network traffic distribution by finding more efficient paths and leads to load balance in the network.

The objective of this research is to address traffic engineering issues in Software-defined Networking (SDN), an innovative method for network architecture and management. Enhancing SDN performance requires prioritizing traffic allocation, avoiding swarm behavior, and optimizing resource utilization. The enhanced Ant Colony Optimization (ACO) method was employed for traffic engineering in a Software-Defined Networking (SDN) network, resulting in enhanced traffic distribution and load balancing. Future research could incorporate heuristic optimization algorithms, such as PSO and GA, to achieve better results.

References

[1] "Traffic classification offloading to stateful data plane in Software-Defined Networking," 2017.

[2] Y. Yu, C. Qian, and X. Li, "Distributed and collaborative traffic monitoring in software defined networks," *Proceedings of the third workshop on Hot topics in software defined networking,* 2014.

[3] U. Shende and V. Bagdi, "A Review on Traffic Engineering Techniques in Software Defined Networks," *2019 International Conference on Intelligent Sustainable Systems (ICISS),* pp. 503-506, 2019.

[4] D. Sanvito, I. Filippini, A. Capone, S. Paris, and J. Leguay, "Clustered robust routing for traffic engineering in software-defined networks," *Comput. Commun.,* vol. 144, pp. 175-187, 2019.

[5] T. Das, V. Sridharan, and M. Gurusamy, "A Survey on Controller Placement in SDN," *IEEE Communications Surveys & Tutorials,* vol. 22, pp. 472-503, 2020.

[6] C. Wu, S. Zhou, and L. Xiao, "Dynamic Path Planning Based on Improved Ant Colony Algorithm in Traffic Congestion," *IEEE Access,* vol. 8, pp. 180773-180783, 2020.

[7] A. Guo and C. Yuan, "Network Intelligent Control and Traffic Optimization Based on SDN and Artificial Intelligence," *Electronics,* 2021.

[8] M. K. Rajoriya and C. P. Gupta, "Sailfish optimization-based controller selection (SFO-CS) for energy-aware multi-hop routing in software defined wireless sensor network (SDWSN)," *International Journal of Information Technology,* vol. 15, no. 7, pp. 3935-3948, 2023.

[9] L. Naibaho, P. R. Saxena, K. Sharma, B. S. Alfurhood, L. Pallavi, and B. Pant, "Integration of Artificial Intelligence in Software Defined Networking Technology through Developing Machine Learning Algorithms," in *2023 3rd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, 2023, pp. 513-517: IEEE.

[10] H. G. A. Elrahim, N. N. N. Abd Malik, and K. B. M. Yousif, "Revolutionizing Load Management in SDIoT Networks: A Multicriteria Approach for SDN Controller," *2023 IEEE Asia Pacific Conference on Wireless and Mobile (APWiMob),* pp. 13-19, 2023.

[11] A. K. Singh, S. Maurya, and S. Srivastava, "Varna-based optimization: a novel method for capacitated controller placement problem in SDN," *Frontiers of Computer Science,* vol. 14, 2019.

[12] A. Filali, Z. Mlika, S. Cherkaoui, and A. Kobbane, "Preemptive SDN Load Balancing With Machine Learning for Delay Sensitive Applications," *IEEE Transactions on Vehicular Technology,* vol. 69, pp. 15947-15963, 2020.

[13] Y. Guo *et al.*, "Traffic Management in IoT Backbone Networks Using GNN and MAB with SDN Orchestration," *Sensors (Basel, Switzerland),* vol. 23, 2023.

[14] R. Kumar, U. Venkanna, and V. Tiwari, "Optimized traffic engineering in Software Defined Wireless Network based IoT (SDWN-IoT): State-of-the-art, research opportunities and challenges," *Computer Science Review,* vol. 49, p. 100572, 2023.

[15] H. Xue, K. T. Kim, and H. Y. Youn, "Dynamic load balancing of software-defined networking based on genetic-ant colony optimization," *Sensors,* vol. 19, no. 2, p. 311, 2019.

[16] H. Zheng, J. Guo, Q. Zhou, Y. Peng, and Y. Chen, "Application of improved ant colony algorithm in load balancing of software-defined networks," *The Journal of Supercomputing,* vol. 79, no. 7, pp. 7438-7460, 2023.

[17] F. Masood, W. U. Khan, S. U. Jan, and J. Ahmad, "AI-enabled traffic control prioritization in software-defined IoT networks for smart agriculture," *Sensors,* vol. 23, no. 19, p. 8218, 2023.

[18] S. Sathyanarayana and M. Moh, "Joint route-server load balancing in software defined networks using ant colony optimization," in *2016 international conference on high performance computing & simulation (HPCS)*, 2016, pp. 156-163: IEEE.

[19] Y. Wei, X. Zhang, L. Xie, and S. Leng, "Energy-aware traffic engineering in hybrid SDN/IP backbone networks," *Journal of Communications and Networks,* vol. 18, no. 4, pp. 559-566, 2016.

[20] Z. Wang, J. Wang, and C. Yan, "An Energy-efficient Traffic Scheduling Method based on Slime Mould Algorithm for SDN," in *2023 6th International Conference on Information Systems and Computer Networks (ISCON)*, 2023, pp. 1-6: IEEE.

[21] L. Cheng, K. Wang, L. Wei, Y. Liang, and P. Song, "A Novel Automatic Voltage Control Strategy Based on Adaptive Pheromone Update Improved Ant Colony Algorithm," in *2022 2nd International Conference on Electrical Engineering and Mechatronics Technology (ICEEMT)*, 2022, pp. 232-236: IEEE.

[22] S. Khan, A. Akram, H. Alsaif, and M. Usman, "Emulating software defined network using mininet-ns3-WIFI integration for wireless networks," *Wireless Personal Communications,* vol. 118, no. 1, pp. 75-92, 2021.

[23] H. Manzoor, S. Manzoor, N. Ali, M. Sajid, M. I. Menhas, and X. Hei, "An SDN-based technique for reducing handoff times in WiFi networks," *International Journal of Communication Systems,* vol. 34, no. 16, p. e4955, 2021.

[24] X. Yizhen, M. Jun, and Z. Ke, "Application of ACO algorithm for searching optimal test link in reconfigurable scanning networks," *Journal of Guilin University of Electronic Technology,* vol. 43, no. 1, pp. 20-26, 2023.

[25] A. Singh, "Swarm intelligence and internet of everything optimization solutions," in *AIP Conference Proceedings*, 2023, vol. 2587, no. 1: AIP Publishing.

[26] A. Umar *et al.*, "A Review on Congestion Mitigation Techniques in Ultra-Dense Wireless Sensor Networks: State-of-the-Art Future Emerging Artificial Intelligence-Based Solutions," *Applied Sciences,* vol. 13, no. 22, p. 12384, 2023.

[27] S. Song and J. Wang, "Research on data center load balancing based on particle swarm optimization fusion ant colony optimization algorithm," in *3rd International Conference on Applied Mathematics, Modelling, and Intelligent Computing (CAMMIC 2023)*, 2023, vol. 12756, pp. 970-975: SPIE.