ENERGY-EFFICIENT TRAFFIC SCHEDULING MECHANISM BASED ON IMPROVED GENETIC ALGORITHM

Ningzhuo BAI[1], Honghui LI[2,*], Xueliang FU[3]

The swift advancements in information technology have ushered in the era of large language models and artificial intelligence. Alongside the rapid expansion of data center scale and business demands, the issue of energy consumption in data center networks has become increasingly prominent. Software-Defined Networking (SDN) revolutionizes network architecture by decoupling the data plane from the control plane, enhancing network flexibility and its adoption in data center networks. To tackle data center energy consumption, this paper introduces an energy-saving traffic scheduling algorithm for SDN data center networks, utilizing an enhanced genetic algorithm. Leveraging network monitoring capabilities in SDN, the algorithm enhances traffic scheduling through an advanced genetic algorithm and puts inactive switches into sleep mode to meet the demands of current traffic flow. This approach reduces active network devices, leading to energy savings. Simulation results confirm the method's feasibility and effectiveness.

Keywords: Data center network; Software defined networking; Genetic algorithm; Energy-saving flow scheduling; Metropolis criterion

1. Introduction

The rapid advancement of information technology, along with the rise of large language models and AIGC, has placed computing power at the forefront of societal focus. Data centers, serving as the backbone of the digital economy, are pivotal in our shift to this new era. By June 2023, China's operational data center racks are anticipated to surpass 7.6 million standard units, boasting a computing power of 197EFLOPS. This achievement ranks China second globally, marking a 30% increase from the same period in 2022. However, with the continuous expansion of computing power, the challenge of energy consumption in data centers has become more acute. Over the past five years, China's data center energy consumption has grown at a rate of 15% annually, with electricity consumption expected to hit 270 billion kW・h in 2023—3.1% of the nation's total. This amount of energy could sustain two mega cities for an entire year [1].

[1] College of Computer & Information Engineering, Inner Mongolia Agricultural University, China
[2] College of Computer & Information Engineering, Inner Mongolia Agricultural University, China, e-mail: lihh@imau.edu.cn
[3] College of Computer & Information Engineering, Inner Mongolia Agricultural University, China

In a data center's equipment makeup, encompassing both peripheral and network devices, it's crucial to recognize that network equipment's energy consumption represents 45%-50% of the data center's total energy use. To maintain the data center network's Quality of Service (QoS), devices like switches are required to offer constant network services, operating at nearly full power irrespective of the actual network load. This leads to considerable energy wastage [2]. Statistics indicate that data center network resource utilization typically falls between 5% and 25% [3].

Data center networks commonly employ the "rich connectivity" network architecture, with Fat-tree [4] being the most popular. This architecture boasts a simple structure, abundant bandwidth resources, and the ability to provide multiple equivalent end-to-end routes, thereby enhancing network redundancy and robustness. However, compared to traditional data center network structures, the "rich connectivity" architecture introduces more network devices, which in turn increases network energy consumption.

Currently, the Equal-Cost Multi-Path (ECMP) routing algorithm [5] is the most commonly utilized network traffic scheduling algorithm in data center networks. When there are multiple equivalent paths to the same destination node, the ECMP algorithm utilizes Hash operation to schedule the traffic. Based on the calculation results, the data flow is evenly distributed to multiple paths to achieve network load balancing. This effectively prevents one or more paths from occupying too much bandwidth and affecting the quality of network service [6]. However, the primary objective of the ECMP algorithm is to achieve network load balancing. When the algorithm evenly distributes the data flow into multiple equivalent paths, it reduces the load of network devices but also increases the use of devices, resulting in low resource utilization and energy consumption waste.

Mckeown proposed the Software Defined Networking (SDN) [7]. By decoupling the network control plane and data plane, network managers can exercise greater flexibility in controlling the network. This, in turn, provides new opportunities for network optimization and energy conservation. With the advent of SDN technology, the control part of the switch is centralized in the controller, while the switch only forwards operations based on the flow table. The data center network can dynamically adjust and optimize itself based on real-time traffic conditions, and the administrator can easily control the sleep or wake state of the switch device port. As a result, software-defined networking has become widely used in data center networks [8].

This paper comprises six chapters. Chapter 1 outlines the background and significance of the IGA-SDN algorithm. Chapter 2 delves into recent related research and details the innovations of the IGA-SDN algorithm. Chapter 3 establishes the mathematical model for the energy-saving traffic scheduling problem and provides a detailed description of the IGA-SDN algorithm's workflow. Chapter 4 presents simulation experiments of the IGA-SDN algorithm and analyzes the results to demonstrate its feasibility. Chapter 5 summarizes the IGA-SDN algorithm, discussing its practical significance, limitations, and potential for future development.

2. Related work

As research into Software Defined Networking (SDN) technology deepens, an increasing number of scholars are integrating SDN into data center network studies. To tackle data center energy consumption, researchers have proposed energy-saving approaches utilizing SDN technology [9,10].

Kurroliya et al. proposed a Gray Wolf Aware Load Balancing and Energy Saving (GLE) algorithm based on edge weight for network optimization [11]. The GLE algorithm employs a hybrid of genetic and Grey Wolf algorithms to identify the optimal network flow solution, harmonizing load balancing with energy efficiency.

Lu et al. proposed a policy-based Minimum Energy Consumption (MEC) heuristic algorithm [12]. MEC takes into account energy saving and quality of service of communication load simultaneously, while ensuring the satisfaction of transmission load. Additionally, they proposed three traffic scheduling strategies. The energy consumption is minimized by selecting the appropriate transmission path while ensuring that each flow satisfies the TCAM size limit and link capacity limit.

Oliverira et al. proposed a heuristic energy-saving strategy [13]. The proposed strategy mainly focuses on the control plane and utilizes the parallel and distributed nature of the controller to improve energy efficiency while maintaining network fault tolerance.

Ibrahim et al. proposed an energy-aware routing Multi-layer Mapping Problem (EARMLP) algorithm for multi-controller SDN networks [14]. The Controller Placement Problem (CPP) is introduced, and a heuristic algorithm is used to perform flow re-selection based on capability awareness to meet the required conditions, ultimately reducing energy consumption. However, determining the locations of all controllers and switches in a real environment is not practical.

Wei et al. proposed a routing algorithm MNLR [15] that utilizes the multinomial Logit model to realize path selection. By calculating the energy of each path, the probability of selecting each path is obtained, and then the path is selected according to the priority of the network flow. The priority depends on the traffic size, meaning that large flows with low priority are migrated to guarantee the delay requirement of small flows.

Zhang proposed the Energy-Saving Algorithm for Software Defined Networks based on Algebraic Connectivity (ESSDNAC) [16] to address the issue of energy consumption. ESSDNAC only requires the parameters of network topology to ensure network connectivity stability. The critical degree is calculated by analyzing the change in algebraic connectivity, and links with small critical degrees are selected to enter the sleep state. However, ESSDNAC does not require real-time network traffic data input, which may cause delays in flow routing and prevent real-time dynamic scheduling routing of flow.

Liu et al. proposed the Correlation Aware Traffic Aggregation (CATC) model based on SDN [17]. The global control ability of SDN was utilized to quantify the correlation of two network flows through Pearson correlation coefficient. The network flows were integrated according to the obtained value to concentrate them into fewer links as much as possible. The rate of the link was set based on the bandwidth utilization of the link to achieve the purpose of link rate adaptation and reduce the number of switches operating at full load to save energy. However, the scalability of the algorithm is limited. As the network size increases, the computational complexity will significantly increase, leading to delays.

Wu et al. proposed an energy-saving algorithm for Internet routing based on traffic matrix (EEBTM) [18], which builds on energy-aware routing. The algorithm calculates paths in the network using the shortest path algorithm and arranges network links in order of bandwidth utilization through network monitoring. After each permutation, the link with the minimum bandwidth utilization is removed from the network topology and put into sleep mode to achieve energy savings while ensuring current network communication. However, the algorithm’s scalability is limited, and its computational complexity increases significantly as the network scale expands, making it difficult to judge whether the network is unblocked.

Energy-efficient traffic scheduling in SDN data center networks presents a highly complex optimization challenge. It requires consideration of multiple network performance indicators, such as link utilization, latency, and load balancing. Additionally, the dynamic nature of network states, including traffic patterns, link conditions, and configurations, further complicates the problem. Traditional algorithms used in data center networks often become impractical due to the high computational cost involved in addressing these challenges. Heuristic algorithms can simplify the problem or use a divide-and-conquer strategy to reduce computational complexity. They also offer strong scalability, making them suitable for large-scale networks. Therefore, this paper introduces a heuristic algorithm to address the energy-efficient scheduling problem. Due to its adaptability, the genetic algorithm can quickly search the feasible domain and has parallel processing capabilities, enabling simultaneous operations on multiple individuals, making it well-suited for network scheduling policy calculations.

This paper presents an energy-saving traffic scheduling algorithm for SDN data center networks, based on an enhanced genetic algorithm. Firstly, for new flows without corresponding flow tables, this work suggests an improved shortest path algorithm via the minimum spanning tree protocol to enhance link reuse and minimize energy consumption in data center networks under low loads; Secondly, the genetic algorithm has been refined by uniformly calculating all data streams to derive a scheduling strategy. Incorporating the Metropolis criterion diminishes the likelihood of converging on local optima, steering the strategy towards a global optimum and further reducing energy consumption; Finally, a link blocking prevention mechanism is introduced to decrease the number of network devices while maintaining service quality.

3. Energy-efficient traffic scheduling mechanism based on improved genetic algorithm

3.1 Mathematical model of energy efficient traffic scheduling problem

This paper develops a mathematical model to address the energy-saving traffic scheduling challenge. The model ensures network flows meet the required bandwidth and optimizes traffic scheduling to reduce active switches and links, enhancing network energy efficiency. The modeling approach is detailed as follows.

The data center network is modeled as a graph Graph(S,L). S represents the set of switches, switches $s_i \in S(1,2,\cdots,|S|)$. L represents the set of links, link $l_i \in L(1,2,\cdots,|L|)$. The maximum bandwidth capacity of each link l is C_l. The links are bidirectional and operate independently. Bandwidth utilization is u_l. Link utilization measures the ratio between the total bandwidth of all data flows through a link and that link's bandwidth capacity. Maximum Link Utilization (MLU) u_{max} represents the maximum link utilization value of the link set L in the data center network. Assuming that all data of each network flow follow a path, the energy consumption of working nodes is denoted as $Cost_s$, the energy consumption of working links is denoted as $Cost_l$, and the nodes and links in sleep state do not generate energy consumption. Therefore, the objective function of energy saving optimization of data center network can be defined as Equation (1).

$$P = Min \sum_{s_i \in S, l_i \in L} \left(\theta_{s_i} \cdot Cost_{s_i} + \theta_{l_i} \cdot Cost_{l_i} \right) \tag{1}$$

Where $\theta_{s_i}, \theta_{l_i} \in [0,1]$, at $\theta_{s_i} / \theta_{l_i} = 1$, indicates that the current switch/link is in the working state at this instant, otherwise it is in the sleeping state.

Define the set of all network flows in the network as E, network flow $e \in E$. L_i and $L_i^{'}$ are the subsets of the link set, the switch s_i^d is the end node of the link

$l_i \in L_i$, and the data flow flows into the switch s_i^d through l_i. s_i^b is the beginning node of $l_i^{'} \in L_i^{'}$, and the data flow passes through $l_i^{'}$ outgoing switch s_i^b. Denote the set of all data flows as E, the bandwidth of a data flow as b_e, and the source and destination switches of a data flow e are denoted by s_e^b and s_e^d, respectively. The variable x_l^e indicates whether the data flow e goes through link l or not. In order to ensure the quality of service of the network and ensure the smooth flow of the network, so as to better achieve the optimization goal, the data center network flow should meet the following two constraints:

$$\sum_{e \in E} x_l^e b_e \le C_l ; l \in L \tag{2}$$

$$\begin{gathered} \sum_{e \in E, l \in L} x_l^e b_e = \sum_{e \in E, l^{'} \in L^{'}} x_{l^{'}}^e b_e ; s_i \in S - \left\{ s_e^b, s_e^d \right\} \\ \sum_{e \in E, l \in L} x_l^e b_e = 0; s_i = s_e^b \\ \sum_{e \in E, l^{'} \in L} x_{l^{'}}^e b_e = 0; s_i = s_e^d \end{gathered} \tag{3}$$

Equation (2) indicates that the current bandwidth used by the link cannot be greater than the maximum bandwidth capacity of the link, so as to ensure that the network is not blocked and reduce the packet loss rate. Equation (3) indicates that if the switch s_i is an intermediate node, the amount of outgoing and incoming data of the switch should be conserved, that is, the incoming data flow of the node is equal to the outgoing and data flow. If the switch s_i is the source switch s_e^b, the data flow will only flow from that node; If the switch s_i is the destination switch s_e^d, then the data flow will only flow in from that node.

3.2 Energy efficient traffic scheduling algorithm IGA-SDN

3.2.1 Overall architecture of energy efficient traffic scheduling algorithm

Fig. 1 depicts the two-layer architecture of the proposed energy-saving scheduling algorithm, featuring the control layer and the data layer. The control layer includes key modules such as network perception, scheduling strategy calculation, energy saving, and flow table generation and distribution. Primarily, the data layer is made up of network devices.

Network Perception Module: This module leverages the network monitoring capabilities of Software-Defined Networking (SDN) to gather network topology information via LLDP within the OpenFlow protocol. Network flow and device port statuses are determined by sending Flow_State_Reply and Port_State_Reply messages and receiving Flow_State_Request and

Port_State_Request messages in return. This module continuously monitors and records network activity, supplying vital information to the scheduling policy calculation module.

Scheduling Policy Calculation Module: This module consists of two sub-modules: the New Flow Policy Calculation Module and the Rescheduling Flow Calculation Module. The New Flow Policy Calculation Module calculates scheduling policies for new flows that the Network Perception Module has not detected in existing flow tables. This paper primarily employs an enhanced shortest path algorithm for calculations. The Rescheduling Flow Policy Calculation Module targets network flows identified by the Network Perception Module as needing rescheduling and employs an optimization algorithm to devise energy-efficient scheduling policies.

Energy saving module: The Energy-Saving Module controls network equipment states, activating necessary switches and links to a “green” state as determined by the Scheduling Strategy Calculation Module, and setting idle ones to a “red” sleep state.

Flow table generation and distribution module: The Flow Table Generation and Distribution Module translates the network flow scheduling policy, as calculated by the Scheduling Policy Calculation Module, into a switch-recognizable flow table format. This table is subsequently dispatched to the appropriate switch for flow forwarding.

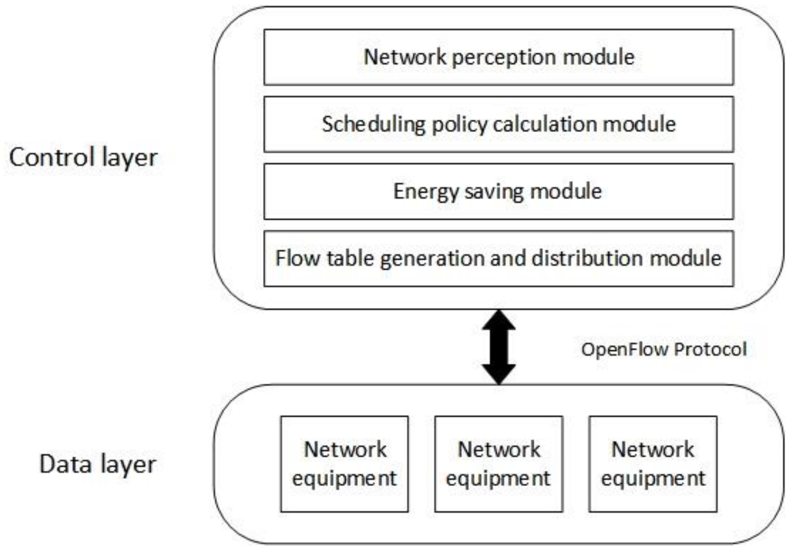

Fig. 1. Overall architecture design of network flow scheduling

3.2.2 Energy-efficient traffic scheduling mechanism

This study proposes an energy-saving traffic scheduling mechanism based on an improved genetic algorithm. Upon encountering a new flow without an existing flow table, the system employs an enhanced shortest path algorithm for scheduling. Flows requiring rescheduling are compiled into a set for which a uniform scheduling policy is calculated. This approach not only optimizes the scheduling strategy but also reduces the optimization time, thereby enhancing the quality of network service.

During optimization, the genetic algorithm is susceptible to converging on local optima. To mitigate this issue, this chapter introduces a temperature parameter and applies the Metropolis criterion, the simulated annealing algorithm's cornerstone, enhancing the algorithm's ability to escape local optima. The steps of the algorithm are illustrated in Fig. 2 and described subsequently.

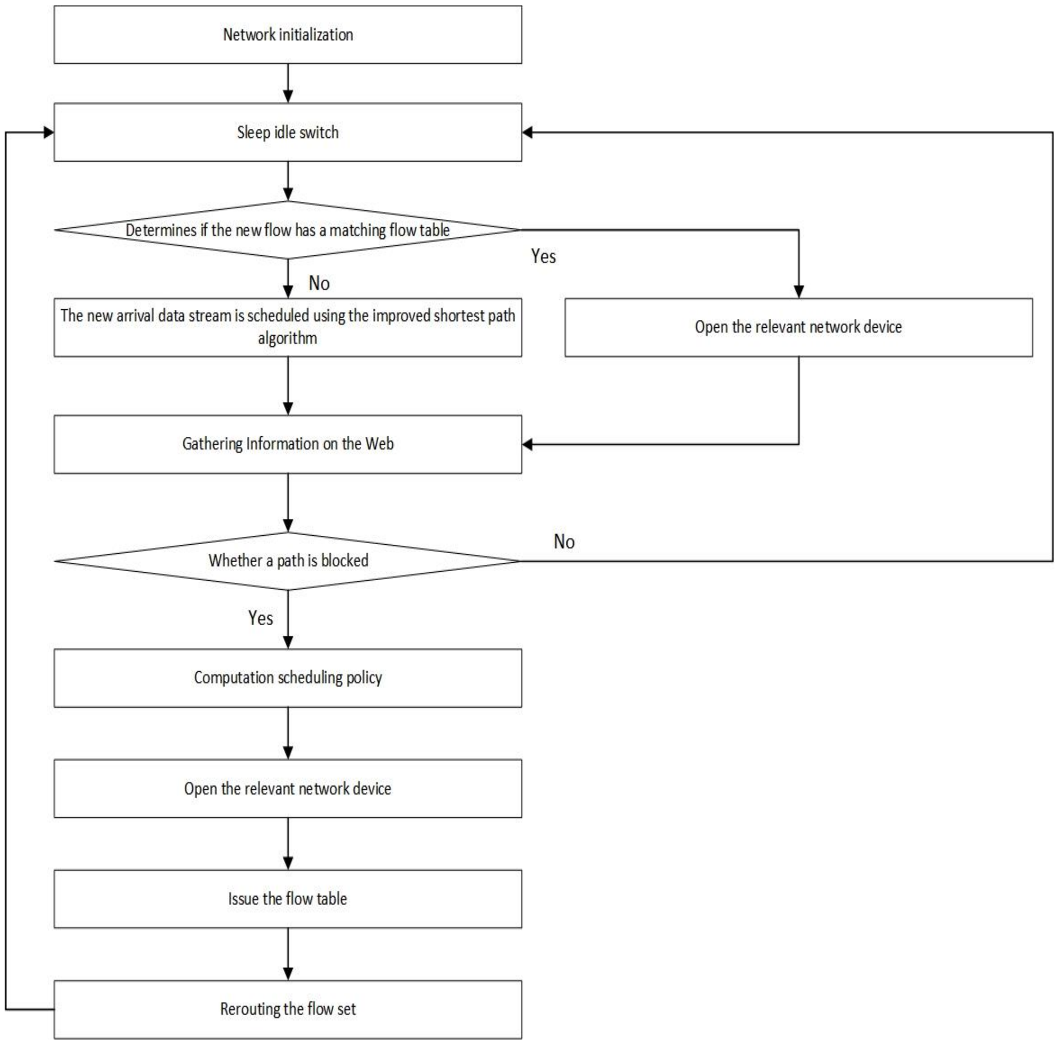

Fig. 2. Flowchart of the scheduling algorithm

(1) Network Initialization: Activate all switches to enable the controller to acquire complete network information.

During network initialization, the controller employs the network awareness module to gather switch location information through the switch_features_handler messages from switches. Additionally, the controller dispatches the table_miss flow table, allowing switches to forward flows even when there’s no matching entry in the flow table. The global awareness capability is leveraged to acquire the complete network topology, facilitating the formulation of the scheduling strategy at a later stage.

(2) Set switches that have received the information to a dormant state. During this phase, active switches and links are shown in green, whereas dormant switches and links are marked in red.

(3) Upon the arrival of a new network flow, the switch initially parses the flow’s information, extracting source and destination addresses and the relevant edge switch port. If no matching entry is found, the switch activates the table_miss flow table entry and forwards the flow information to the controller. The controller calculates the network flow’s forwarding path using an enhanced shortest path algorithm, based on the entire network topology. It marks the forwarding path’s switches and links green, transforms the path into flow table entry format, and dispatches it as the default forwarding table to the relevant switches. If a matching flow entry exists, the switch activates the relevant switches and links, utilizing the entry for network forwarding operations.

The enhancement of the shortest path algorithm aims to reduce the number of active network devices and maintain seamless connectivity throughout the network while devising scheduling strategies for new flows. For instance, in a Fat-Tree topology with k=4, featuring four core switches, one core switch serves as the root. The nodes in the resultant minimum spanning tree form the node set for the enhanced shortest path algorithm, which then exclusively selects nodes from this set as forwarding nodes for new flows. The conventional ECMP algorithm identifies multiple equivalent paths and employs a hash operation to randomly select among these paths. Consequently, flows sharing the same source and destination may traverse different paths, leading to a greater number of switches being engaged. The enhanced shortest path algorithm guarantees that flows with identical sources and destinations follow a singular path, thus minimizing the number of switches utilized across the network and maintaining link connectivity.

(4) Leveraging SDN’s global monitoring capabilities, the switch persistently sends Flow_State_Request and Port_State_Request messages via the network awareness module, receiving Flow_State_Reply and Port_State_Reply messages in return. Network flow size is determined by analyzing the Flow_State_Reply message, and current bandwidth utilization for network device ports and links is ascertained through the Port_State_Reply message. Using this information, idle ports and links are designated red. If all port on a switch is red, the entire switch is marked as red. Flows exceeding the port’s available bandwidth

are flagged for rescheduling and proceed to step (5). Otherwise, the switch persists in sending Flow_State_Request and Port_State_Request messages for network surveillance and to track its flow table entries. Should the flow table entry differ from the default—derived using the improved shortest path algorithm—the high-priority entry is removed. Finally, the highest priority of the default flow table entry is restored.

(5) After compiling a list of network flows requiring scheduling, the controller inputs the corresponding paths into the enhanced genetic algorithm to determine the overall scheduling strategy. The optimization algorithm outputs a scheduling strategy, indicating the forwarding path for each network flow after scheduling. The controller then processes the scheduling policy, activates the necessary network devices, transforms the policy into a flow table format, and dispatches it to the appropriate switch.

(6) Continue to step 3.

3.2.3 Energy-efficient traffic scheduling strategy based on improved genetic algorithm

The detailed process of IGA-SDN algorithm are as follows:

Step 1 Initialization of the algorithm

Initialize the parameters iter and Max_iter. Set the number of variations mu_num, the initial temperature T_0 is, the termination temperature T_{end}, the current temperature T, and the cooling rate α.

(2) Take the incoming set of required scheduling flows size Num as the population size and the set of paths as the initial population Pop, where each individual Pi is a complete available path, i=1,2,... ,Num. Each individual Pi consists of n links l as shown in the equation.

(3) The fitness function is defined as Equation (4), and the formula for judging whether to accept a new individual is defined as Equation (5).

$$F\left(X_i\right) = sw_{cur\_path} - sw_{cur\_path} \cap sw_{working} \tag{4}$$

$$P\left(\Delta E\right) = e^{-\frac{\Delta E}{T}} \tag{5}$$

Where sw_{cur_path} is the link matrix used by the current path, and 1 in the matrix means that the link connected by two switches is in use, for example, sw_{cur_path}[1][2]=1 means that the link connected by switches with Dpids 2 and 3 is used in the current path. $sw_{working}$ represents the link matrix among all jobs in the current network. sw_{cur_path} intersects $sw_{working}$ to indicate the number of working links used in the current path, and swcur_path subtracts the intersection, that is, the number of links that need to be opened if the current path is used. Obviously, the smaller the fitness function is, the more beneficial it is to achieve energy saving. is the value of the original individual energy minus the new individual energy, and T is the current temperature.

Step 2 Optimize the operation

Genetic algorithm optimization utilizes crossover and mutation operations. For each individual Pi in the population Pop, a crossover operation is conducted, and mutation operations target specific individuals.

1. Crossover operation. The KSP algorithm is used to select a path P_c with the same origin and destination as individual P_i for crossover operation, that is, the two individuals are traversed, the first node of the two individuals is found, and the later nodes are exchanged to form a new individual $P_C^{'}$. According to the formula, individual Pi and $P_C^{'}$ fitness were calculated respectively, and the Metropolis algorithm was called to select two individuals if $\Delta E < 0$, $P_C^{'}$ instead of individual P_i to enter the next generation population. If $\Delta E > 0$, it is calculated $P(\Delta E)$ according to Equation (5) and a random number r of (0,1) is generated; if $r < P(\Delta E)$, the new individual $P_C^{'}$ is retained to the next generation population instead of individual P_i; otherwise, the original individual P_i is retained.

2. Mutation operation. Select mu_num individuals P_{m_i} with the minimum bandwidth utilization in the population for mutation operation, where i=1,2,... , mu_num. It traverses all the links in individual P_{m_i}, selects the link with the minimum link utilization, and replaces it with link mi to form the new individual $P_{m_i}^{'}$ after mutation, where m_i is the path with the highest bandwidth utilization among multiple paths generated by KSP algorithm, and can make the mutation path $P_{m_i}^{'}$ as a feasible solution after replacement. Calculate the fitness of P_{m_i} and $P_{m_i}^{'}$, and call the Metropolis algorithm to select two individuals, calculate ΔE, if $\Delta E < 0$, $P_{m_i}^{'}$ replaces individual P_{m_i} into the next generation population. If $\Delta E > 0$, calculate $P(\Delta E)$ according to Equation (7) and generate a random number r of (0,1). If $r < P(\Delta E)$, the new individual $P_{m_i}^{'}$ is retained in the next generation population instead of individual P_{m_i}; otherwise, the original individual P_{m_i} is retained.

Step 3 Cooling operation. The original temperature is T_1, the current temperature $T = T_1 \cdot \alpha$.

Step 4 determines whether the iteration end condition is reached, if not, jump to step 2; Otherwise, the iteration is terminated, the current scheduling policy is output, and the switches and links are turned on or off by the controller, and the flow table is sent to the relevant switches.

4. Simulation experiment and result analysis

This paper assesses the IGA-SDN energy-saving algorithm's performance by comparing it with two algorithms: ECMP and GFF. ECMP, the most widely used traffic scheduling algorithm in data centers, randomly selects a path based on the hash values of multiple equivalent paths [6]. GFF, a globally preferred algorithm, chooses the first available path with enough bandwidth to accommodate the current flow based on its size [19] .

4.1 Calculation method of network energy consumption proportion

This study employs the network energy consumption ratio and maximum link utilization as performance metrics for the algorithm. The network energy consumption ratio, defined as the proportion of energy consumed by active switches and links to the total energy, is detailed in Equation (6).

$$Network\_energy\_consumption\_ratio = \frac{\sum_{s\in S,l\in L}\left(\theta_s \cdot Cost_s + \theta_l \cdot Cost_l\right)}{\sum_{s\in S,l\in L}\left(Cost_s + Cost_l\right)} \tag{6}$$

Here, a variable value of $\theta_s / \theta_l = 1$ indicates that the switch or link is active, while any other value signifies a dormant state. $Cost_s$ and $Cost_l$ represent the energy consumption of a single switch/link, and the specific values are shown in Table 1 [20]. Under the same load, the lower the network energy consumption ratio is, the higher the energy efficiency of the network is, and the better the energy saving effect is.

Table 1

Energy consumption of network equipment/W

Hierarchy	Switch/W	Link/W
Core layer	176	2.18
Edge layer	135	1.80

4.2 Experimental environment

This research leverages the Mininet simulation platform and the Python programming language to develop a Fat-tree network topology with a branching factor of k=4. The network comprises 20 switches, all adhering to the OpenFlow protocol, and includes 16 hosts, each pair connected to an access layer switch. All network links have a bandwidth capacity set at 10Mb/s. For effectiveness assessment of the proposed algorithm, a Ryu controller running on Python 3.0 is utilized.

4.3 Experiment with network traffic patterns

Due to the limited availability of open-source data in SDN data center networks, which often involves industry-specific privacy concerns, and the highly variable, complex, and uneven traffic distribution of actual data center networks, this paper uses a custom-developed version of the Iperf traffic generation tool on the Mininet simulation platform. Three traffic modes are generated [21], where the flow size follows an exponential distribution, the flow duration follows a Poisson distribution, and the source and destination nodes are randomly selected based on predefined rules, simulating data center traffic more accurately [22]. The three traffic modes are as follows:

(1) Random pattern. The source and destination hosts are randomly selected in the network.

(2) Interval mode Stride(x). Each host hi sends data to host h(i+x) mod n, where n is the total number of hosts in the network. In this experiment, n is 20 and x is 2.

(3) The pattern Staggered(p1,p2). Each host transmits data with probability p1 to hosts belonging to the same access layer switch. Transmit data to hosts belonging to the same Pod with probability p2; Send data to the hosts of other pods with probability (1-p1-p2).

In this study, data stream sizes follow an exponential distribution, while data stream durations adhere to a Poisson distribution. Each experiment lasted 10 minutes and was conducted 20 times. Network energy consumption data were collected from the central five-minute period of each trial and averaged to serve as the effective experimental data.

4.4 Experimental method

The experimental steps used in this paper are shown in Figure 3 and described later.

(1) Initialize the Mininet network simulation environment and establish a connection with the Ryu controller.

(2) Generate traffic: Simulated network traffic is produced in Mininet based on the rules outlined in Section 4.3.

(3) Select the source and destination hosts. Based on the three traffic modes detailed in Section 4.3, choose the source and destination hosts on the network and initiate simulated traffic from the source to the destination host.

(4) Obtain traffic and port information. The Ryu controller monitors and gathers network traffic data, as well as real-time status updates of network device ports.

(5) Calculate relevant network parameters. The Ryu controller computes link utilization using the collected traffic and port data.

(6) Check if the current duration has reached 10 minutes. If not, go to Step 2; otherwise, proceed to Step 7.

(7) The Ryu controller will calculate and output network parameter statistics.

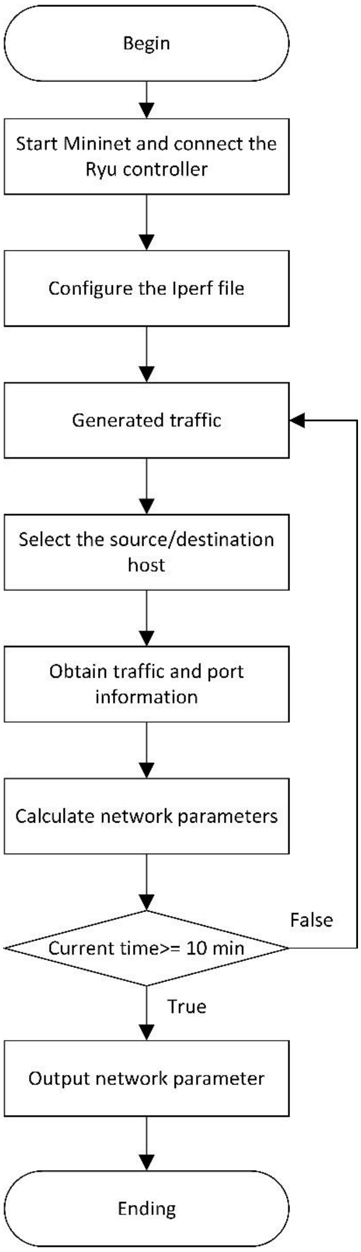

Fig. 3. Experimental flow chart

4.5 Experimental related parameter Settings

As a heuristic algorithm, the setting of control parameters directly influences its results. Focused on energy-efficient traffic scheduling, this study selects the main parameters, as detailed in Table 2, based on findings from literature

[23, 24] and multiple simulation experiments, to configure the experiment's relevant settings.

Table 2

Main parameter values of DE-SASDN energy-saving algorithm

Parameters	Value
$Maxiter$	30
M	20
cf	0.3
T_0	100
T_d	10
α	0.92

Through several experimental tests, the number of iterations is selected as 30 and the final temperature is selected as 100 degrees in this paper. In this paper, the upper limit of the path between two points is set as 11 hops. In the Fat-tree network topology with k=4, the possibility that the path more than 11 hops is the optimal solution is very small, so the upper limit of the optimal solution number of the optimization algorithm in the experimental network is set as 10.

4.6 Experimental results and analysis

4.6.1 Network energy consumption ratio

Given the vast and complex data flow in data center networks, this study opts for three communication modes for simulation experiments. Network energy consumption proportion is determined by using Equation (6). To validate the experimental data's authenticity, this study tests three traffic scheduling algorithms across the three communication modes. In this experiment, each algorithm undergoes 20 trials, with each lasting 10 minutes. Data from the central five-minute segment are collected and averaged to constitute the experiment's final outcome.

(1) Random pattern

In the Random traffic mode, the simulation traffic scheduling of IGA-SDN algorithm, GFF algorithm and ECMP algorithm is carried out respectively. As shown in Fig. 4, compared with GFF and ECMP algorithms, IGA-SDN reduces the network energy consumption by 3.75%～46.63% and 3.00% ~38.20%, respectively.

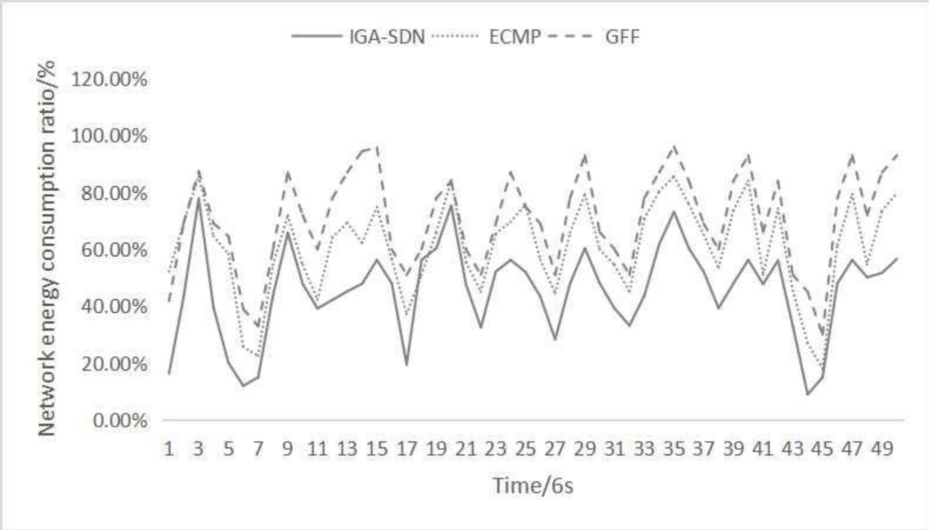

Fig. 4. Proportion of network energy consumption in Random mode

(2) Stride mode

In the Stride(x) traffic mode, x is set to 2, and the simulation traffic scheduling is carried out for IGA-SDN algorithm, GFF algorithm and ECMP algorithm respectively. As shown in Fig. 5, compared with GFF and ECMP, IGA-SDN reduces the network energy consumption by 4.31%～52.52% and 1.69%～38.98%, respectively.

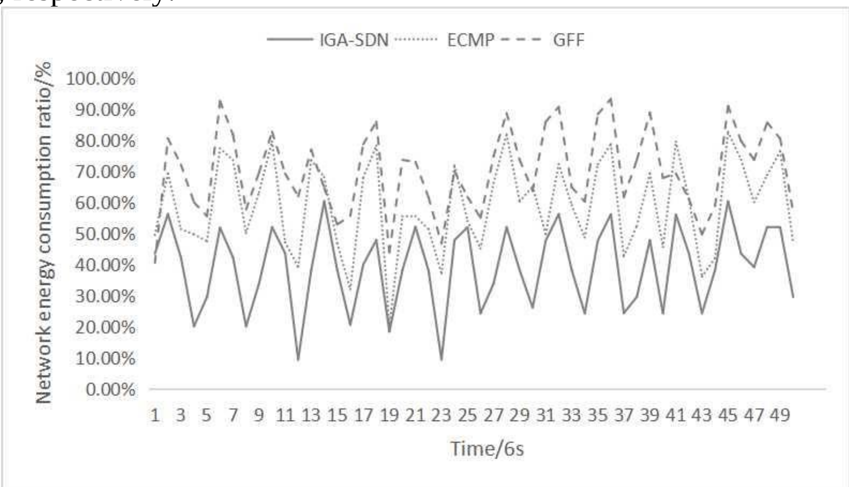

Fig. 5. Proportion of network energy consumption in Stride mode

(3) Starggered pattern

In Starggered(p1,p2) mode, p1=0.2 and p2=0.4 are taken in this paper. After several experiments for each algorithm separately, the results are shown in Fig. 6. It can be seen from the figure that compared with DE-SASDN and ECMP algorithm, IGA-SDN has a certain reduction in the proportion of network energy consumption, which is decreased by 4.86%-48.96% and 0.18%-40.08% respectively.

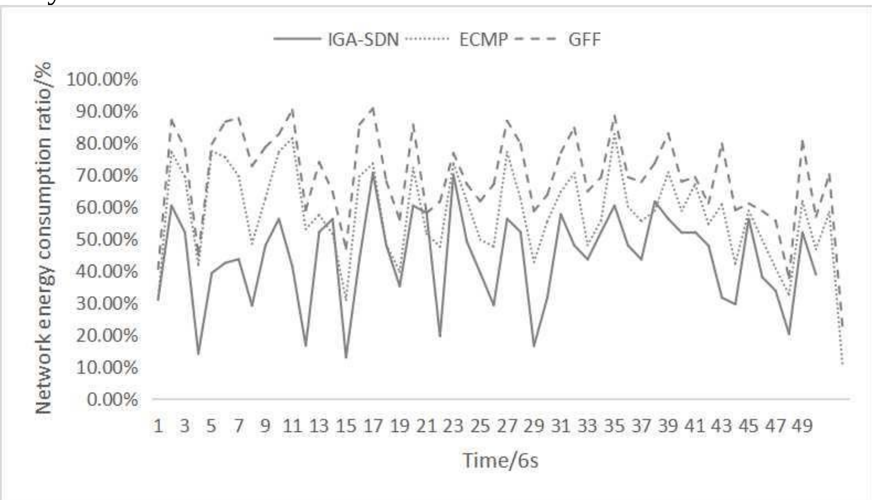

Fig. 6. Proportion of network energy consumption in Starggered mode

4.6.2 Maximum link utilization of the network

To comprehensively assess the algorithm's performance, this study conducts a comparative analysis of three algorithms based on the network's maximum link bandwidth utilization.

Maximum link bandwidth utilization refers to the usage of the most heavily loaded link within the network. To save energy, idle network devices are set to a sleep state, concentrating network flow on as few links as possible for forwarding. Thus, with a similar overall network traffic level, higher maximum link bandwidth utilization indicates more concentrated flow and more devices in sleep mode. Conversely, a lower utilization signifies more devices in active state. Therefore, the maximum link utilization rate can partially indicate the algorithm's energy-saving efficiency.

In order to prevent network congestion caused by increasing link bandwidth utilization and resulting in network fluctuations, the following operations are done in this paper:

(1) When the optimization algorithm selects the optimal solution, it will compare the current flow size with the residual bandwidth of the path to ensure the feasibility of the scheduling strategy.

(2) Set the buffer bandwidth. When calculating the residual bandwidth of the path, the maximum bandwidth of the path will be set to 95% of the original bandwidth. Although there are elephant flows in the network, the traffic accounts for a large proportion, but the number of flows is relatively small. Setting 5% of the buffer bandwidth can basically meet the scheduling of all network flows.

This experiment selects three algorithms of Random mode for experiments and presents the experimental results of maximum link utilization in the form of cumulative distribution function diagram. A point in a cumulative distribution function (CDF) plot represents the proportion of the total data that is less than the abscissa of the point. The longer the distance between two adjacent points is, the more concentrated MLU of the algorithm is in this region.

As shown in Fig. 7, the MLU of the three algorithms in Random mode is compared. It can be seen that the maximum link utilization is mainly concentrated between 85% and 95%. However, ECMP selects the shortest path when selecting the path, so the maximum bandwidth utilization is improved, which is concentrated between 65% and 85%. The MLU of IGA-SDN is mainly concentrated between 85% and 95%, which is greatly improved compared with ECMP and GFF.

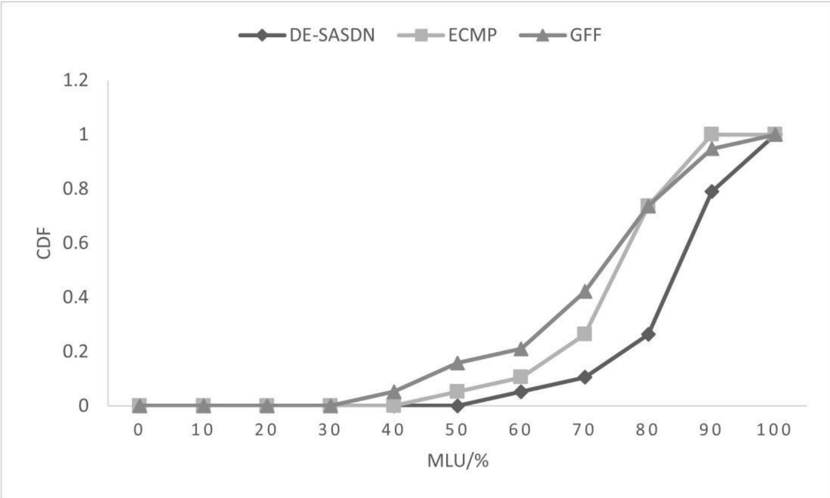

Fig. 7. Maximum link utilization

In summary, by comparing the network energy consumption ratios and the maximum link bandwidth utilizations in Random mode across three algorithms, it is demonstrated that the IGA-SDN algorithm significantly enhances network link utilization and reduces the network’s energy consumption ratio, all while maintaining service quality, thus effectively saving energy.

5. Summary

This study introduces an energy-efficient SDN traffic scheduling algorithm utilizing an enhanced genetic algorithm. Leveraging network states gathered by the controller, the enhanced genetic algorithm computes an overall scheduling strategy for flows requiring scheduling, thereby minimizing the number of active network devices for energy conservation. The simulation results demonstrate that the proposed algorithm reduces network energy consumption by up to 52.52%

compared to the GFF and ECMP algorithms, while maintaining quality of service. This indicates that the proposed algorithm effectively achieves energy savings.

All research and experiments in this study are conducted in a single-controller environment, for which the Ryu controller is particularly well-suited. As the network scale expands, a single controller's control and load capacities may no longer suffice. Consequently, future research will focus on expanding experimental scales, exploring multi-controller environments, and enhancing the optimization algorithm's control precision.

Acknowledgement

This work is partially supported by the National Natural Science Foundation Project (61962047), the Inner Mongolia Autonomous Region Natural Science Foundation Project (2024MS06002), the Inner Mongolia Autonomous Region Science and Technology Major Special Project (2021ZD0005), the Inner Mongolia Autonomous Region Directly Affiliated Universities Basic Research Business Fee Project (BR22-14-05), the Inner Mongolia Autonomous Region University Science and Technology Innovation Team Construction Special Project (BR231302), the Inner Mongolia Autonomous Region Science and Technology Plan Project (2022YFHH0070).

R E F E R E N C E S

[1]. *Chen Xiaopeng, Yue Xin, Liang Yudong*: Research on Artificial intelligence promoting green energy saving in data centers. Information and Communication Technology and Policy, 2024, 50(02): 33-39.

[2]. *Chen Xin-Tuo, Zhou Li-Yang, Zhang Cheng-Bin, et al*. Research status and development trend of heat dissipation and cooling technology for green and energy efficient data center. Engineering Science, 2022, 24(4): 94-104.

[3]. *Heddeghem WV, Lambert S, Lannoo B, Colle D, Pickavet M, Demeester P*: Trends in worldwide ICT electricity consumption from 2007 to 2012. Computer Communications 2014, 50(sep.1): 64-76.

[4]. *Al-Fares M, Loukissas A, Vahdat A*. A scalable, commodity data center network architecture. ACM SIGCOMM computer communication review, 2008, 38(4): 63-74.

[5]. *Zhang H, Guo X, Yan J, et al*. SDN-based ECMP algorithm for data center networks. 2014 IEEE Computers, Communications and IT Applications Conference. IEEE, 2014: 13-18.

[6]. *Li Dan, Chen Guihai, Ren Fengyuan, et al*. Research progress and trends of data center networks. Chinese Journal of Computers, 2014, 37(002): 259-274. (in Chinese.)

[7]. *Prabha C, Goel A, Singh J*. A survey on sdn controller evolution: A brief review. 2022 7th International Conference on Communication and Electronics Systems (ICCES). IEEE, 2022: 569-575.

[8]. *Zuo Qing-Yun, Chen Ming, Zhao Guang-Song, et al*. Research on SDN technology based on OpenFlow. Journal of Software, 2013, 24(5): 1078-1097.

[9]. *Niu T, Li H*. A Survey of Energy-Efficient Methods in Data Center Networks Based on Software Defined Network. 2021 IEEE Conference on Telecommunications, Optics and Computer Science (TOCS). IEEE, 2021: 477-483.

[10]. *Wang Jiye, Zhou Biyu, Zhang Fa, et al*. Survey on energy consumption models and Energy efficiency algorithms for Data centers. Journal of Computer Research and Development, 2019, 56(8): 1587-1603.
[11]. *Kurroliya K, Mohanty S, Kanodia K, et al*. Grey wolf aware energy-saving and load-balancing in software defined networks considering real time traffic. 2020 International Conference on Inventive Computation Technologies (ICICT). IEEE, 2020: 689-694.
[12]. *Lu Z, Lei J, He Y, et al*. Energy optimization for Software-Defined data center networks based on flow allocation strategies. Electronics, 2019, 8(9): 1014.
[13]. *Oliveira T F, Silveria L F Q*. Distributed SDN controllers optimization for energy saving. 2019 Fourth International Conference on Fog and Mobile Edge Computing (FMEC). IEEE, 2019: 86-89.
[14]. *Ibrahim A A Z, Hashim F, Sali A, et al*. A multi-objective routing mechanism for energy management optimization in SDN multi-control architecture. IEEE Access, 2022, 10: 20312-20327.
[15]. *Wei M, Zhou J, Gao Y*. Energy efficient routing algorithm of software defined data center network. 2017 IEEE 9th International Conference on Communication Software and Networks (ICCSN). IEEE, 2017: 171-176.
[16]. *Zhang Meiqiong*. Energy Saving Algorithm for Software Defined Network Based on Algebraic Connectivity. Journal of Fujian Normal University (Natural Science Edition), 2021, 37(03): 20-28.
[17]. *Liu Liang, Zhang Lin, Yang Liu, et al*. Correlation-aware traffic consolidation algorithm in Data center networks. Computer Engineering and Applications, 2019, 55(24): 62-67+74.
[18]. *Wu Yong, Geng Haijun*. Research on Energy-efficient routing based on Traffic matrix. Microelectronics & Computer, 2019, 36(3): 82-86.
[19]. *Al-Fares M, Loukissas A, Vahdat A*. A scalable, commodity data center network architecture. ACM SIGCOMM computer communication review, 2008, 38(4): 63-74.
[20]. *Kun X, Xiaohong H, Maode M, et al*. Power savings in software defined data center networks via modified hybrid genetic algorithm. The Journal of China Universities of Posts and Telecommunications, 2017, 24(4): 76-86.
[21]. *Escudero-Sahuquillo J, Garcia P J, Quiles F J, et al*. A new proposal to deal with congestion in InfiniBand-based fat-trees. Journal of Parallel and Distributed Computing, 2014, 74(1): 1802-1819.
[22]. *Deng Gang, GONG Zhenghu, WANG Hong.* Research on modern data center network characteristics. Computer research and Development,2014, 51(2): 395-407.
[23]. *Ding Qingfeng, Yin Xiaoyu*. A Survey of Differential Evolution Algorithms. Journal of Intelligent Systems, 2017, 12(4): 431-442.
[24]. *Dai R, Li H, Fu X*. Elephant flow scheduling in sdn data center network based on differential evolution algorithm. UPB Scientific Bulletin Series C, 2022, 84(2): 221-236.