Journal of Computer and Communications, 2024, 12, 94-104
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

Multipath Selection Algorithm Based on Dynamic Flow Prediction

Jingwen Wang, Guolong Yu, Xin Cui*

School of Computer Science and Technology, Shandong University of Technology, Zibo, China
Email: *wjw12001200@163.com

How to cite this paper: Wang, J.W., Yu, G.L. and Cui, X. (2024) Multipath Selection Algorithm Based on Dynamic Flow Prediction. *Journal of Computer and Communications*, **12**, 94-104.
https://doi.org/10.4236/jcc.2024.127007

Received: June 11, 2024
Accepted: July 23, 2024
Published: July 26, 2024

Abstract

Traditional traffic management techniques appear to be incompetent in complex data center networks, so proposes a load balancing strategy based on Long Short-Term Memory (LSTM) and quantum annealing by Software Defined Network (SDN) to dynamically predict the traffic and comprehensively consider the current and predicted load of the network in order to select the optimal forwarding path and balance the network load. Experiments have demonstrated that the algorithm achieves significant improvement in both system throughput and average packet loss rate for the purpose of improving network quality of service.

Keywords

Data Center Network, Software Defined Network, Load Balance, Long Short-Term Memory, Quantum Annealing Algorithms

1. Introduction

With the development of cloud computing, 5G and other technologies, under the trend of explosive traffic growth, increasing scale and complex topology in Data Center Network (DCN) [1], the traditional network management architecture gradually exposes some deficiencies. Therefore, designing more effective and intelligent load balancing algorithms is of great research significance to improve the network performance in cloud data centers.

The proposal of SDN [2] provides strong technical support for the effective detection and management of networks. SDN separates the control plane from the data plane, so that the data plane is only responsible for routing and forwarding, and the control plane implements forwarding decisions, realizing universal forwarding and efficient manipulation of network data streams, thus increasing the flexibility of the network.

We first use machine learning to dynamically predict the network flow and design a multipath forwarding policy based on the current and predicted network load characteristics to dynamically adjust the network load situation. The effectiveness of this algorithm is verified by comparing it with other load balancing algorithms through simulation experiments.

2. Related Work

In recent years, SDN-based network load balancing technology has received widespread attention because of its high flexibility and adjustability, and researchers at home and abroad have conducted a series of studies on it and proposed a variety of solutions.

Fu Wang [3] and others proposed a Dynamic Distributed Multi-path (DDMP) load balancing algorithm, which quickly responds and adjusts the traffic according to the inverse of the buffer occupancy when certain paths in the network are loaded to prevent severe congestion. However, as the scale of the data center network expands, its energy consumption will become an important issue. M. A. Saifullah [4] *et al*. proposed EHLBOF (Extended Health monitoring for Load Balancing in OpenFlow based network) algorithm, which not only takes into account the current load of the servers, but also their health status, thus improving service availability and response speed. In addition, Zhang Chaohui [5] *et al*. proposed an energy efficient routing algorithm based on bandwidth matching, which is able to reserve sufficient space for the upcoming data streams, preventing link congestion and saving network energy consumption.

A. Montazerolghaem [6] proposed a modular energy and load control system in IoMT that both reduces the number of loMT active servers and switches and balances the load distribution. Literature [7] proposed a network control mechanism based on SDN and AI, and proposed three operator network optimization algorithms, which provide effective solutions and theoretical support for telecom operators to implement intelligent network control and traffic optimization in SDN.

The MTDLR [8] algorithm integrates path-level metrics to select forwarding paths and adapts to different topologies by adjusting the weights of network influences. However, MTDLR determines the weighting parameter to select the optimal path only through the average flow bandwidth utilization, which may cause the algorithm to fall into a local optimal solution and is not flexible enough for the dynamically changing network load problem.

3. Load Balancing Algorithm Based on Dynamic Flow Prediction and Multi-Constraint Routing

3.1. Dynamic Flow Forecasting

3.1.1. Long Short-Term Memory

The LSTM model [9] was proposed by Hochreiter and Schmidhuber in 1997, which effectively solves the problem of gradient vanishing and gradient explo-

sion encountered when dealing with long sequential data. The structure of LSTM is shown in Figure 1.

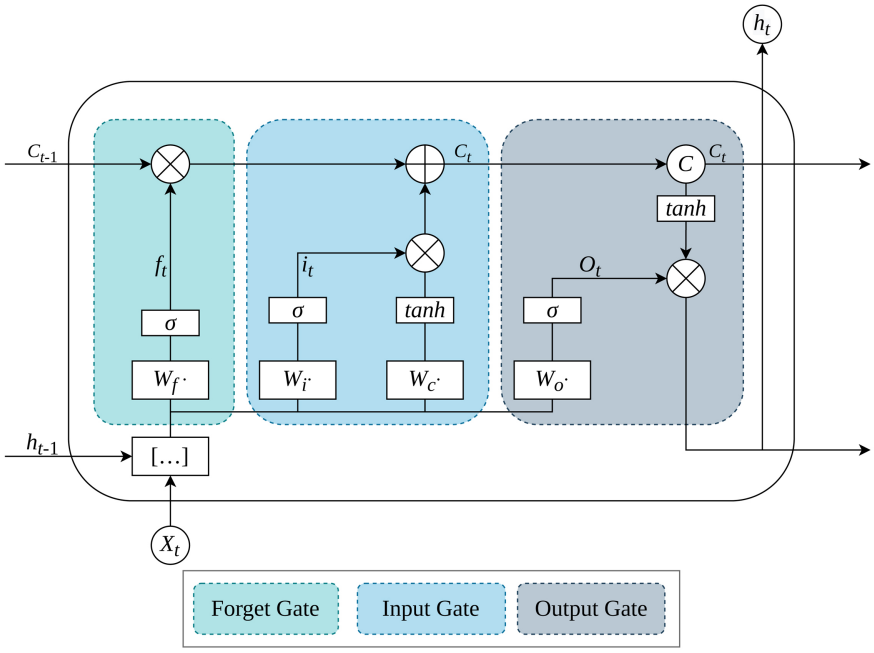

Figure 1. LSTM structure.

The LSTM model adds three parts to the RNN (Recurrent Neural Network), namely, forgetting gate, input gate and output gate, and each gating unit consists of σ and tanh activation function.

1) Forgetting gate: determines which old information is forgotten. $f(t)$ indicates the importance of the old information, the higher the value, the more important the information.

$$f(t)=\sigma\left(W_f\left[h_{t-1},x_t\right]+b_f\right) \tag{1}$$

2) Input gate: i_t decides which new information is stored. is the output of the input gate, indicating the importance of the new information. $\tilde{C}_i$ indicates new information can be stored to the current cell state.

$$i_t=\sigma\left(W_i\left[h_{t-1},x_t\right]+b_i\right) \tag{2}$$

$$\tilde{C}_t=\tanh\left(W_c\left[h_{t-1},x_t\right]+b_c\right) \tag{3}$$

At this point, the cell state is updated and the updated state is C_t.

3) Output Gate: Determines what information will be output.

$$C_t=f_tC_{t-1}+i_t\tilde{C}_t \tag{4}$$

$$h(t)=\sigma\left(W_o\left[x_t,h_{t-1}\right]+b_o\right) \tag{5}$$

In Equations (1) to (4), W is the network weight coefficient, h_{t-1} is the output of the previous time step, h_t is the output of the current time, and b is the bias of the activation value.

3.1.2. Model Evaluation Indicators

The average absolute error (MAE), root mean square error (RMSE), average absolute percentage error (MAPE) and Symmetric Mean Absolute Percentage Error (SMAPE) are usually used to evaluate the prediction effect of the model, and the lower the value of the indexes, the more stable the model is and the more accurate the prediction results are. The formulas are as follows:

$$\mathrm{MAE} = \frac{1}{m}\sum_{t=1}^{m}\left|\left(h_t - \hat{h}_t\right)\right| \tag{6}$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{m}\sum_{t=1}^{m}\left(h_t - \hat{h}_t\right)^2} \tag{7}$$

$$\mathrm{MAPE} = \frac{100\%}{m}\sum_{t=1}^{m}\left|\frac{\hat{h}_t - h_t}{h_t}\right| \tag{8}$$

$$\mathrm{SMAPE} = \frac{100\%}{m}\sum_{t=1}^{m}\frac{\left|h_t - \hat{h}_t\right|}{\left|h_t\right| + \left|\hat{h}_t\right|} \tag{9}$$

where m, h_t, and $\hat{h}_t$ are the total number of samples, measured values, and predicted values, respectively. MAE measures the average size of the absolute value of the difference between the predicted and actual values; RMSE is the square root of the mean of the squares of the difference between the predicted and actual values; MAPE is the average percentage of the absolute value of the ratio of the prediction error to the actual value; SMAPE reduces the problem of numerical inflation that may occur with MAPE by taking the absolute value of the predicted value and the actual value and averaging them.

3.2. Quantum Annealing Algorithms

Metropolis *et al*. [10] proposed the idea of Simulated Annealing (SA) in 1953, which has since been widely used in combinatorial optimization [11] problems. SA, as a heuristic global optimization algorithm, simulates the physical process of atoms rearranging to reach thermodynamic equilibrium during annealing of solid materials. Although SA can handle complex optimization problems, its convergence speed and and solution quality depend on the parameter settings, and the temperature of the object is directly proportional to the energy, at lower temperatures, according to the Metropolis criterion, it only accepts new solutions that are smaller than the current solution, or when the temperature decreases at too fast a rate, it may skip many potential globally optimal solutions, and thus easily causes the system to stay at the local optimal solution.

With the development of quantum technology, Quantum Annealing (QA) algorithm [12] based on quantum tunneling effect is proposed. The so-called quantum tunneling effect, *i.e.*, the quantum leap gives the quantum the ability to penetrate the potential barrier higher than its own energy. Simulated annealing algorithm and quantum annealing algorithm over the potential barrier are shown in **Figure 2**, the particle reaches the local optimum at point A, the simu-

lated annealing algorithm needs to go over the potential barrier in order to achieve the global optimum, and the quantum annealing algorithm only needs to reach the global optimum point B directly through the quantum tunneling effect.

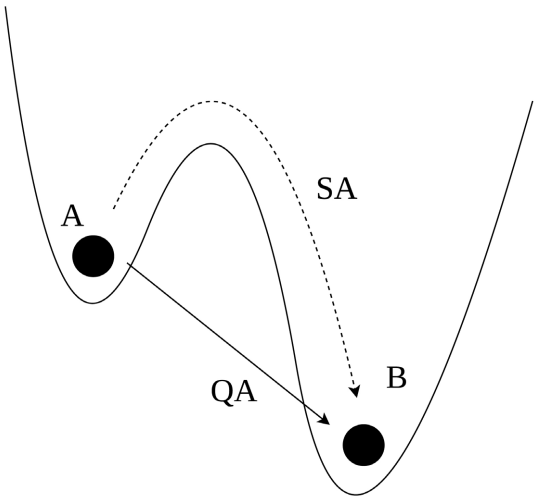

Figure 2. Comparison of SA and QA.

The process of realizing quantum annealing requires 6 steps:

Step 1: Construct the evaluation function $H_q = H_{pot} + H_{kin}$ of the quantum system, where H_{pot} is the potential energy to represent the objective function of the optimization problem and H_{kin} is the kinetic energy;

Step 2: Set the initial temperature T, the transverse magnetic field strength Γ, and the maximum number of iterations *Steps*. Select an initial state x randomly at temperature T and calculate $H_q(x)$;

Step 3: Generate a new state x' by randomly perturbing state x and calculate its $H_q(x')$;

Step 4: Calculate $\Delta H_q = H_q(x) - H_q(x')$. If $\Delta H_q > 0$, or $\Delta H_q < 0$ and $e^{-\frac{\Delta H_q}{T}} < \text{random}(0,1)$, the new state x' is accepted, otherwise, return to step 3;

Step 5: Perform a de-tempering and magnetic field attenuation operation;

Step 6: Determine whether the maximum number of iterations or the termination condition is reached, if so terminate, otherwise return to step 3.

According to the above steps, the flowchart of quantum annealing algorithm is shown in **Figure 3**:

3.3. Multi-Constraint Based Path Forwarding Policy

After a data stream is generated, there are multiple paths between its source and destination nodes in the data center, and choosing the appropriate path for forwarding can effectively maintain network load balancing. In this paper, we design the optimal forwarding path policy that satisfies the constraints through the current and predicted load balancing degree.

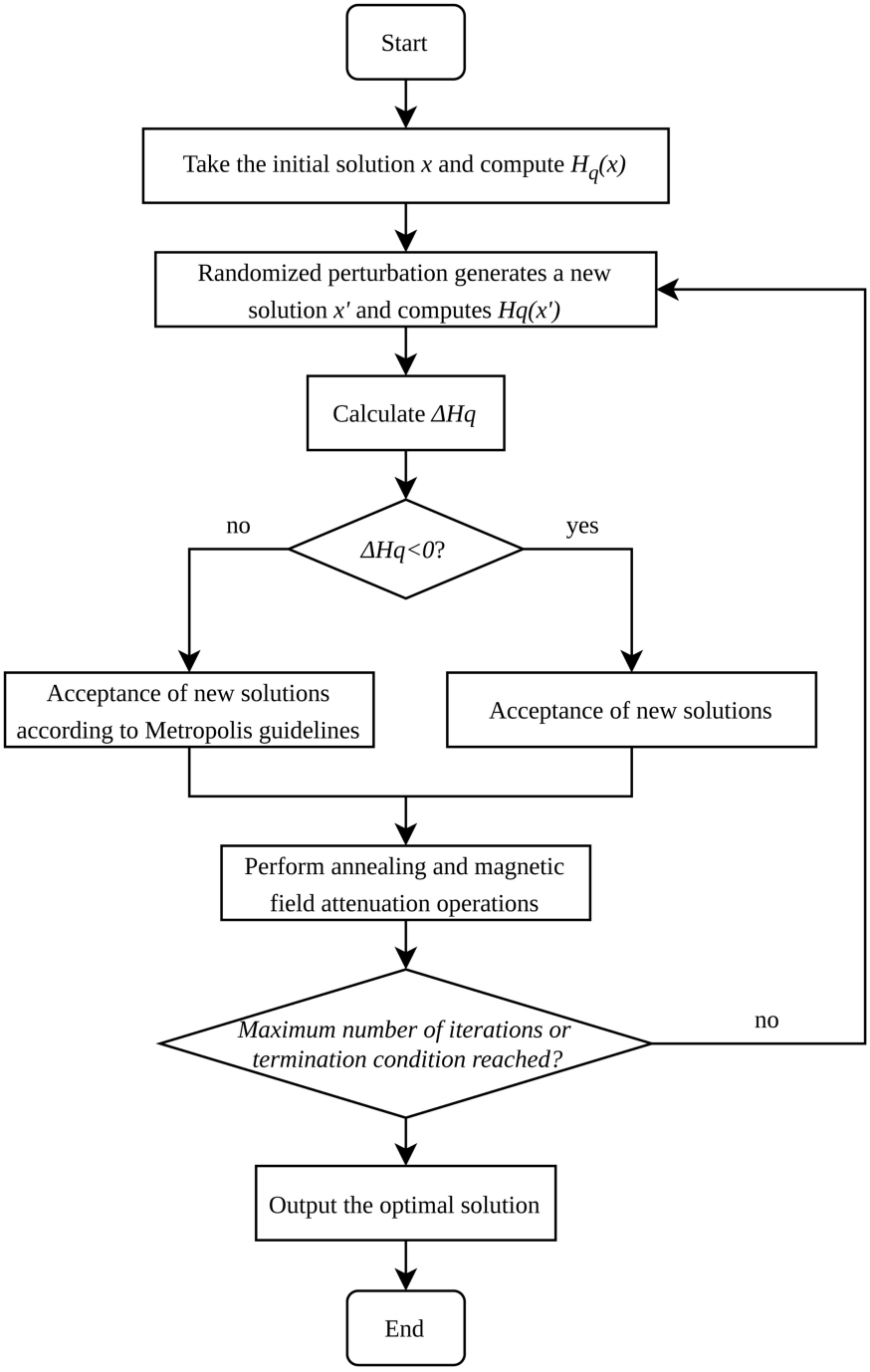

Figure 3. Flowchart of QA.

Construct the data center network as a directed graph $G=(S,L)$, with S being the set of all nodes in the network and L being the set of all links in the network. Denote the path between the data flow source node s and the destination node d as $P_{s,d}$.

3.3.1. Path Available Bandwidth

In a data center network, choosing a path each time to forward with the maxi-

mum available bandwidth between source and destination nodes can effectively alleviate and prevent load imbalance problems. Get the switch port statistics to get the number of bytes sent and received by the port and calculate the transmission rate $speed_{i,j}$ of the port based on the last recorded number of bytes transmitted:

$$speed_{i,j} = \frac{stats_{i,j,t} - stats_{i,j,t-1}}{pd} \tag{10}$$

In Equation (10), i and j respectively denote the source and destination switches, *stats* denotes the number of bytes transmitted on the port, and *pd* denotes the time interval between two recordings.

Then the bandwidth bw_l of link l is denoted as the minimum of the two ports connected, and the available bandwidth $free\_bw$ is the difference between the link capacity C_l and the used bandwidth, and the minimum available bandwidth of all the links in the path is denoted as the available bandwidth of the path:

$$bw_l = \min\left(speed_{i,j}, speed_{j,i}\right) \tag{11}$$

$$free\_bw_l = C_l - bw_l \tag{12}$$

$$A = \min\left(free\_bw_{l_1}, free\_bw_{l_2}, \cdots, free\_bw_{l_n}\right) \tag{13}$$

3.3.2. Link Load Balancing Degree

A link is often part of multiple paths, so the available bandwidth of different links in a path varies. To avoid overloading certain links and improve transmission efficiency, the load balancing degree of a link is considered when selecting a forwarding path. In this paper, the bandwidth equalization degree of a path is expressed as the difference between the minimum and maximum values of the bandwidth between all nodes in the path:

$$B = \frac{\min\left(free\_bw_{l_1}, free\_bw_{l_2}, \cdots, free\_bw_{l_n}\right)}{\max\left(free\_bw_{l_1}, free\_bw_{l_2}, \cdots, free\_bw_{l_n}\right)} \tag{14}$$

3.3.3. Link Forwarding Policy

Combining the available bandwidth and bandwidth equalization of the above paths with the LSTM prediction of the available traffic and bandwidth equalization, the transmission cost of the paths is used as a weight for path selection. In addition, in order to guarantee the transmission of the data flow, one path is selected at a time and only one path is selected, and there are no loops in the selected paths, so that the objective function definition is obtained:

$$\begin{aligned} &\min \frac{1}{A + B + A_{pre} + B_{pre}} \\ &\text{s.t.} \begin{cases} \sum_{r=1}^{n} x_r \le 1 \\ P = p \end{cases} \end{aligned} \tag{15}$$

In Equation (15), n is the number of all forwarding paths calculated from the source destination address.

3.4. Design of LSTM-QALB Path Forwarding Algorithm

The optimal path is solved by quantum annealing algorithm and the solution process is shown in Algorithm 1:

Algorithm 1. Optimal path selection based on LSTM and QA.

Input: topological directed graph G
Output: optimal forwarding path

1. Get the set of forwarding paths all_paths based on the source directory address.
2. **for** path in all_paths **do**
3. Calculate the available bandwidth for all links on the path, save to total_weights.
4. Predict the available bandwidth of all links on the path, saved to total_forecast_weights.
5. **for** weights in total_weights **do**
6. Calculate the available bandwidth of the path and save it to free_bws.
7. Calculate load balancing of paths, save to balances.
8. **for** forecast_weights in total_forecast_weights **do**
9. Calculate available bandwidth for path prediction, save to forecast_free_weights.
10. Calculate the load balancing degree of the path prediction and save it to forecast_balances.
11. Define the Hamiltonian quantity.
12. Calculation of optimal forwarding paths based on quantum annealing algorithm.

4. Experimental Analyses

In this paper, we simulate the changes in the distribution of network traffic in the data center through python and choose a Fat-tree with $k = 4$ as the experimental topology, as shown in Figure 4, which contains 4 core switches, 8 aggregation switches, 8 edge switches and 16 servers.

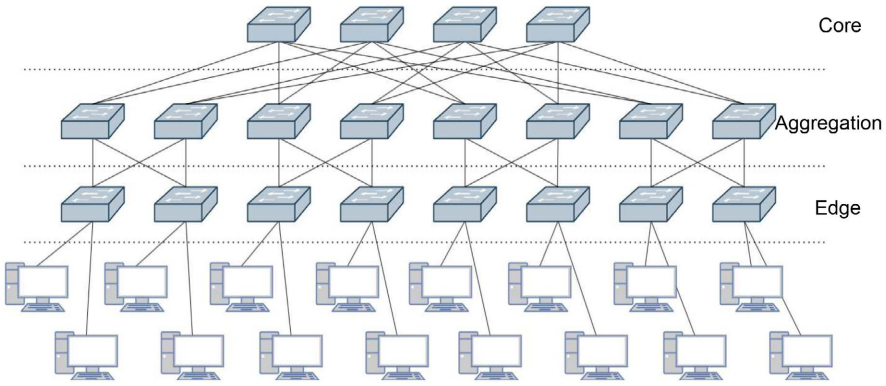

Figure 4. Fat-tree ($k = 4$) topology.

The link bandwidths are all set to 100 Mbps, the link delay is 1 ms, and the traffic data of 64 links are collected from the simulation environment. After a number of learning training, set the number of iterations to 250, using Adam's algorithm, the learning rate is 0.001. Then the traffic is predicted, and four evaluation index values are obtained: MAE = 4.99, RMSE = 7.28, MAPE = 5.65%, SMAPE = 2.76%, the model prediction is more accurate.

The Poisson model [13] is used, where the source and destination hosts are randomly selected each time, so that the number n of packets arriving in time sequence t satisfies the Poisson distribution with parameter λt, *i.e.*, $P(n)=\frac{e^{-\lambda t}(\lambda t)^n}{n!}$, and its corresponding sequence of packet arrivals at the time interval T is exponentially distributed, *i.e.*, $F(T)=1-e^{-\lambda t}$. The parameters were adjusted from 0.1 to 0.9 to simulate different network loads, and each set of experiments was repeated 10 times to improve the accuracy of experimental data.

In this paper, the LSTM-QALB algorithm is compared with MTDLR and EERA algorithms for throughput and packet loss. The experimental results in **Figure 5** and **Figure 6** show that as the network load increases, the system throughput and packet loss rate both increase; when the parameter is greater than 0.5, namely, when the load is high, the system throughput increases slowly and the packet loss rate rises sharply. When the load is low, the performance of the three algorithms is similar, but when the load is high, the LSTM-QALB algorithm outperforms the other two.

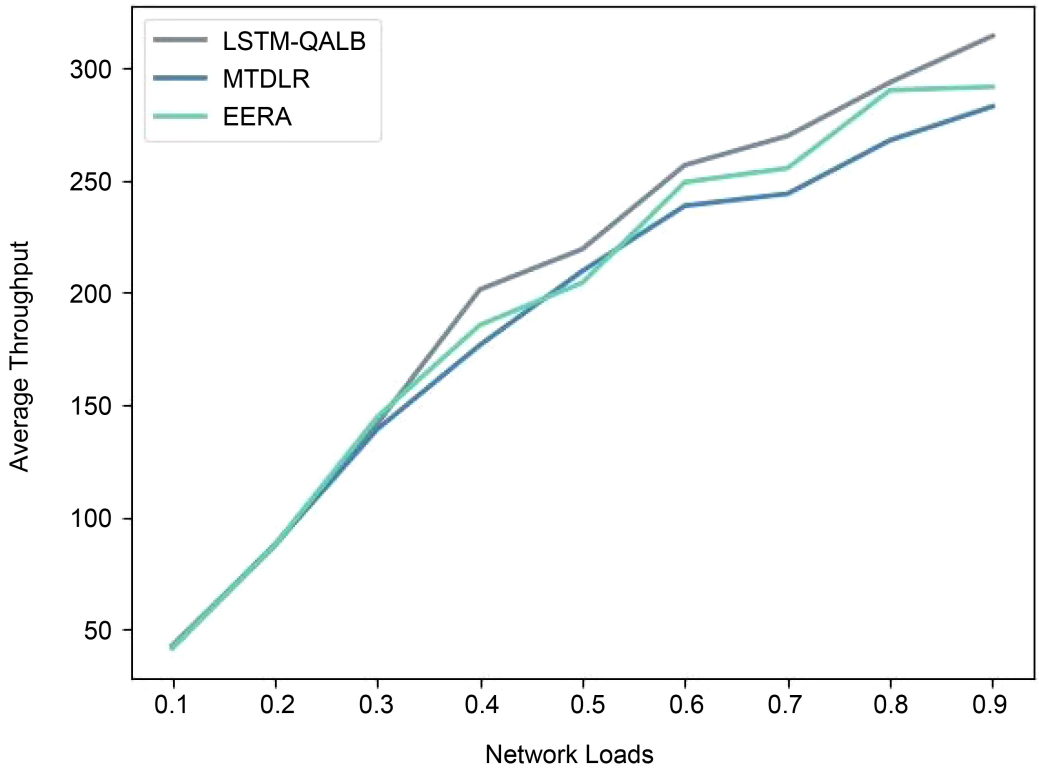

Figure 5. Comparison of system throughput of different algorithms.

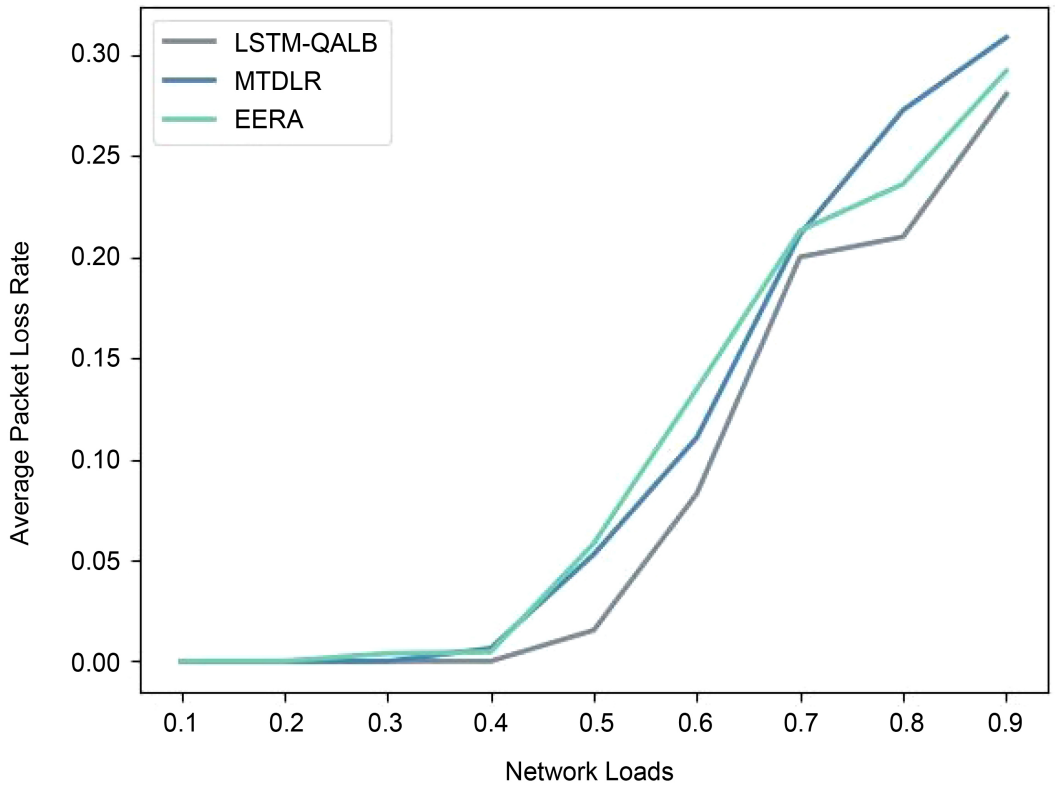

Figure 6. Comparison of average packet loss rate of different algorithms.

5. Conclusion

In this paper, we propose a load balancing algorithm LSTM-QALB based on LSTM dynamic traffic prediction and quantum annealing, which selects the optimal forwarding path by calculating the available bandwidth of the current and predicted paths with the link bandwidth equalization. Compared with MTDLR and EERA algorithms, LSTM-QALB has obvious advantages in terms of system throughput and average packet loss rate, which achieves the purpose of load balancing and improving the network quality of service.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Li, W.X., Qi, H., Xu, R.H., Zhou, X.B. and Li, K.Q.Q. (2020) Data Center Network Flow Scheduling Progress and Trends. *Journal of Computing*, **4**, 600-617.

[2] Macedo, D.F., Guedes, D., Vieira, L.F., Vieira, M.A. and Nogueira, M. (2015) Programmable Networks—From Software-Defined Radio to Software-Defined Networking. *IEEE Communications Surveys & Tutorials*, **17**, 1102-1125. https://doi.org/10.1109/COMST.2015.2402617

[3] Wang, F., Yao, H., Zhang, Q., Wang, J., Gao, R., Guo, D. and Guizani, M. (2021) Dynamic Distributed Multi-Path Aided Load Balancing for Optical Data Center

Networks. *IEEE Transactions on Network and Service Management*, **19**, 991-1005. https://doi.org/10.1109/TNSM.2021.3125307

[4] Saifullah, M.A. and Mohamed, M.M. (2016). Open Flow-Based Server Load Balancing Using Improved Server Health Reports. 2016 2*nd International Conference on Advances in Electrical*, *Electronics*, *Information*, *Communication and Bio-Informatics*, Chennai, 27-28 February 2016, 649-651. https://doi.org/10.1109/AEEICB.2016.7538369

[5] Zhang, C.H. and Zhou, J.Q. Energy-Efficient Scheduling Scheme for Software-Defined Data Center Network Traffic Based on Bandwidth Matching. *Systems Engineering and Electronics*, 1-13. https://link.cnki.net/urlid/11.2422.TN.20240325.1505.018

[6] Montazerolghaem, A. (2022) Software-Defined Internet of Multimedia Things: Energy-Efficient and Load-Balanced Resource Management. *IEEE Internet of Things Journal*, **9**, 2432-2442. https://doi.org/10.1109/JIOT.2021.3095237

[7] Guo, A. and Yuan, C. (2021) Network Intelligent Control and Traffic Optimization Based on SDN and Artificial Intelligence. *Electronics*, **10**, Article 700. https://doi.org/10.3390/electronics10060700

[8] Guo, L. (2018). Designing and Implementation of Load Balancing Methods in Data Center Networks Based on Openflow. Master's Thesis, Beijing Institute of Technology.

[9] Hochreiter, S. and Schmidhuber, J. (1997) Long Short-Term Memory. *Neural Computation*, **9**, 1735-1780. https://doi.org/10.1162/neco.1997.9.8.1735

[10] Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H. and Teller, E. (1953) Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, **21**, 1087-1092. https://doi.org/10.1063/1.1699114

[11] Kirkpatrick, S., Gelatt, Jr. and Vecchi, M.P. (1983) Optimization by Simulated Annealing. *Science*, **220**, 671-680. https://doi.org/10.1126/science.220.4598.671

[12] Finnila, A.B., Gomez, M.A., Sebenik, C., Stenson, C. and Doll, J.D. (1994) Quantum Annealing: A New Method for Minimizing Multidimensional Functions. *Chemical Physics Letters*, **219**, 343-348. https://doi.org/10.1126/science.220.4598.671

[13] Zhang, B., Yang, J.H. and Wu, J.P. (2011) Survey and Analysis on the Internet Traffic Model. *Journal of Software*, **22**, 115-131. https://doi.org/10.3724/SP.J.1001.2011.03950