



A Sub-flow Adaptive Multipath Routing Algorithm for Data Centre Network

You Lu^{1,2} · Yuhao Chen^{1,2} · Xiao Xu^{1,2} · Qiming Fu^{1,2} · Jianping Chen^{1,2} · Lanhui Liu³

Received: 29 August 2022 / Accepted: 7 February 2023
© The Author(s) 2023

Abstract

In response to the fast and intensive traffic changes and the presence of redundant links in the network topology in data centre networks, multipath routing has become the dominant approach. Existing multipath routing algorithms are still inadequate in terms of adapting to dynamic changes in the network state quickly and the cooperation of path selection and sub-flow assignment. Therefore, this paper proposes a sub-flow adaptive multipath routing algorithm for data centre networks (SAMP). The deep deterministic policy gradient (DDPG) algorithm is introduced. DDPG combines deep learning (DL) and reinforcement learning (RL) to implement different network state changes quickly, especially in the process of topology, to achieve dynamic migration of the optimised decisions that have been learned. An adaptive multipath routing model for subflows is established to accomplish collaborative scheduling of path selection and subflow assignment based on the real-time state of the network. The experimental results show that the algorithm can be well adapted to the data centre network, and when compared with several traditional methods, it reduces the latency by an average of 31.3%, improves the task transmission success rate by an average of 14.9%, and increases the throughput by 37.1%.

Keywords Multipath routing · Reinforcement learning · Data center network · Traffic schedule · Software defined network

Abbreviations

DDPG Deep deterministic policy gradient
DL Deep learning
RL Reinforcement learning
DCN Data centre network

DJSP Dijkstra's shortest path
ECMP Equal cost multipath
WCMP Weight cost multipath
DEFT Dynamic error free transmission
SHR Software-defined hybrid routing
MLF Flow characteristics algorithm
DRL Deep reinforcement learning
SAMP Sub-flow Adaptive Multipath Routing
SDN Software defined network

✉ Jianping Chen
alanjpchen@aliyun.com

You Lu
luyou@usts.edu.cn

Yuhao Chen
2113041045@post.usts.edu.cn

Xiao Xu
xuxiao@post.usts.edu.cn

Qiming Fu
fqm_1@126.com

Lanhui Liu
liulanhui1111@126.com

¹ Suzhou University of Science and Technology, Suzhou, China

² Jiangsu Province Key Laboratory of Intelligent Building Energy Efficiency, Suzhou, China

³ Chongqing Industrial Big Data Innovation Center Co., Ltd., Chongqing 400707, China

1 Introduction

In recent years, emerging applications such as streaming media, online classrooms, and network disks have gradually started to be deployed in the cloud, which also places higher performance demands on the data centre network (DCN), which is the infrastructure for cloud computing and storage [1]. Currently, however, routing policies based on Dijkstra's shortest path (DJSP) are so fixed that routing paths have a high repetition rate, which makes data centre network topologies often suffer from link redundancy [2]. Compared to traditional single-path routing, multipath routing makes better use of network resources. Since different

transmission tasks require different numbers of sub-flow paths and the actual conditions vary between paths, the number of subflows, path selection, and allocation ratios are the focus of current research on multipath routing in data centre networks.

In traditional data centre networks, there exist multipath routing such as equal cost multipath (ECMP) [3, 4] algorithm, weight cost multipath (WCMP) algorithm [5], dynamic error free transmission (DEFT) algorithm [6]. These multipath routing algorithms often depend on the static link state for path selection and subflow allocation decisions, and lack the awareness of the dynamic characteristics of the flows. They cannot fully utilise the free links to share the network load [7, 8]. Then some researchers consider the dynamic characteristics of flows in the sub-flow allocation process, such as software-defined hybrid routing (SHR) algorithm [9] and multipath routing on link real-time status and flow characteristics algorithm [10] (MLF), although the dynamic characteristics of long and short flows have been taken into account and multipath transmission is optimised through focused scheduling. However, this approach is still inadequate for sensing dynamic changes in the network state, and will not allow sub-path selection and allocation from a global perspective, thereby affecting multipath transmission performance.

In recent years, with the development of machine learning algorithms, various approaches based on machine learning algorithms have emerged to solve routing problems. Cheng et al. [11] proposed a RL-based routing algorithm that can choose paths in a limited way. However, all flows are handled uniformly in routing. Rischke et al. [12] proposed QR-SDN, which is a scheme based on reinforcement learning. Xu et al. [13] used the Q-routing algorithm to design a multipath routing method. However, there are limitations in the highly dynamic and time-varying environment. With the introduction of deep reinforcement learning (DRL), combining the perceptual capabilities of deep learning (DL) with the decision making capabilities of reinforcement learning (RL) in a generic form. DRL has shown significant capabilities in dealing with highly dynamic time-varying environments and complex state spaces. Fu et al. [14] used Deep Q-learning to generate optimal routing paths for SD-DCN, but there was a lack of cooperation in both subpath selection and sub-flow assignment.

In summary, in the dynamic changing environment of the data centre network state, these multipath routing algorithms are not yet able to simultaneously collaborate on subpath selection, sub-flow assignment and adjust the paths used for multipath transmission adaptively based on the changing network state in real time. The concrete performances are as follows:

(1) Existing algorithms lack real-time updates on the network state and path changes. For different transmission tasks

in different network states, it is necessary to select the right number and structure of sub-paths based on path changes, to improve network performance to the maximum extent.

(2) Existing algorithms are not yet taking into account the combination of the path itself and the subflow allocations. As a result, the sub-stream allocation and scheduling cannot fully cooperate with the changes in path selection, which limits the performance of multipath transmission.

Multipath routing is essentially to select the appropriate path among the multiple reachable paths between the source node and the sink node and allocate sub-streams for parallel transmission [15]. So, it can be transformed into optimal control tasks. Considering the tight connections between network states, routing decisions and the inadequacies of existing work, this paper proposes a sub-flow adaptive multipath routing algorithm for data centre networks (SAMP). Aiming at the routing characteristics of fast and intensive traffic changes and redundant links in the network topology in data centre networks, this paper introduces the deep reinforcement learning method DDPG, which combines deep learning and reinforcement learning. It can quickly achieve changes of different network states, especially in the process of data centre network changes represented by the topology, and can achieve dynamic migration of the already learned optimisation decisions. A sub-flow adaptive multipath routing model is also established to complete collaborative scheduling of path selection and subflow assignment based on the real-time state of the network. Using the centralised control and consistent state view of the network provided by the software defined network (SDN) as the basis for deployment implementation, the algorithm is deployed with the software defined network to learn routing policies and achieve dynamic adjustments by interacting with the network environment. Compared with traditional algorithms, SAMP has a greater improvement in indicators such as transmission delay and throughput. The main contributions of this paper are as follows:

(1) Aiming at the real-time network state change problem, this paper introduces deep reinforcement learning algorithm to adaptively select the appropriate number and structure of sub-paths according to the network status, and dynamically update the routing strategy through feedback and the reward optimisation strategy.

(2) Aiming at the lack of coordination between sub-flow allocation and path selection, this paper establishes a sub-flow adaptive multipath routing model on the basis of reinforcement learning. This allows the algorithm to take into account the distribution of sub-streams when selecting the optimal path, so as to make the full use of the advantages of multipath transmission.

The rest of this paper is arranged as follows: the second part introduces the problem description and related concepts; the third part introduces the multipath routing algorithm and

the deployment implementation framework; the fourth part introduces the experimental settings and results analysis; the fifth part summarises the full text and outlook future research directions.

2 Problem Description and Related Concepts

2.1 Problem Description

Some recent studies have shown that not all streams are suitable for multipath transmission. Therefore, in a certain network state, the performance of multipath routing is related to the number of sub-paths and the distribution ratio of sub-streams.

Taking the classic topology Fat-tree [16] of the data center network in Fig. 1 as an example, the forwarding rate of the intermediate node $N_i(0 \leq i \leq 12)$ is a_i MB/s, and the to-be-forwarded traffic size of the node N_i is X_i MB. $t_i = X_i/a_i$ is the queuing delay of the task at node i . To simplify the path selection, this paper devises hierarchical relationships for the exchange nodes. Then the multipath calculation can avoid taking paths with retracements in the hierarchy. So among the Fat-tree, between node A and node B , there are eight paths like A, N_1, N_5, N_3, B ; $A, N_1, N_5, N_{11}, N_6, N_3, B$ and so on. For the transfer task $Task$ of size S from node A to node B , S/a_i is the sending delay of the task at node i . Parallel transmission is performed using paths A, N_1, N_5, N_3, B and A, N_1, N_6, N_3, B . The parallel transmission delay is $T_1 = \max(t_1 + t_5 + t_3 + S_1/a_1 + S_1/a_5 + S_1/a_3, t_1 + t_6 + t_3 + S_2/a_1 + S_2/a_6 + S_2/a_3)$, where $S = S_1 + S_2$. Single-path transmission using paths A, N_1, N_6, N_3, B . The single-path transmission delay is $T_2 = t_1 + t_6 + t_3 + S/a_1 + S/a_6 + S/a_3$. By comparing T_1 and T_2 , it can be found that the effect of using different

numbers of paths for transmission is largely affected by the states of nodes $A, N_1, N_2, N_3, N_4, N_5, N_6$. Similarly, when selecting paths A, N_1, N_5, N_3, B and A, N_1, N_6, N_3, B for parallel transmission, distributing more sub-streams to a relatively idle path can greatly improve the transmission effect. Therefore, from these two points, the problems to be solved in this paper can be described as follows: we should design an adaptive multipath routing algorithm with real-time network status and transmission tasks as inputs. The algorithm outputs a reasonable multipath routing strategy, which can select a reasonable number of sub-paths and sub-stream allocation ratio for different transmission tasks to improve network transmission performance.

2.2 Related Concepts and Model

When using reinforcement learning method to solve practical problems, it is necessary to abstract the problem scene and transform it into a model that reinforcement learning method can understand. In this paper, the actual application scenario is network routing, which will be modelled in this section.

For the classic data centre network topology Fat-tree as shown in Fig. 1. The network links, forwarding nodes, sending and receiving terminals can be represented by an undirected graph $G = (V, E, c(v), m, n)$. In graph G , $V = \{v_1, v_2, \dots\}$ is the node set in the network, representing transport layer equipment, which is composed of edge layer, core layer and convergence layer. E is the set of edges in the network, representing links, $c(v_i)$ represents the capacity of forwarding nodes v_i in set V , and m and n represent the number of edges and nodes in graph G , respectively. For the convenience of discussion, an adjacency matrix with n rows and n columns $D = [D_{ij}]_{n \times n}$ is used to represent the undirected graph G , as shown in formula (1). When node i and node j are neighbour nodes, both D_{ij} and D_{ji} are 1, otherwise they are 0. Similarly, the communication between nodes in the network is represented by a traffic matrix $M = [m_{ij}]_{x \times x}$ in the form of formula (1), where x is the number of edge layer nodes, and m_{ij} is the traffic from the source node v_i to the target node v_j . When there is no communication volume between the source-sink node, the corresponding communication volume will be set to 0.

$$D = \begin{bmatrix} D_{11} & \dots & D_{1n} \\ \vdots & \ddots & \vdots \\ D_{n1} & \dots & D_{nn} \end{bmatrix}. \tag{1}$$

In this paper, the routing model is constructed based on reinforcement learning. Reinforcement learning, as an important branch of machine learning, is different from supervised learning and unsupervised learning. It continuously interacts with the environment through agent, and can learn the

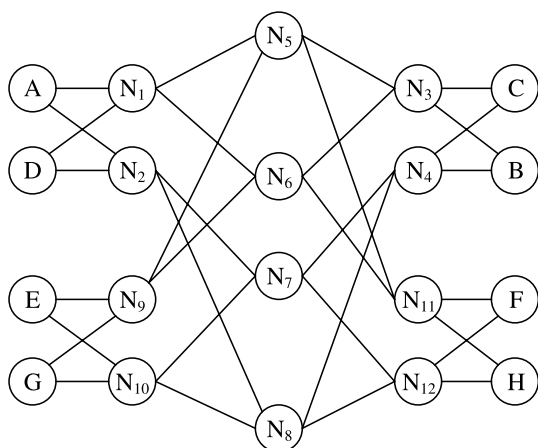


Fig. 1 Data centre network Fat-tree topology

optimal strategy for solving problems online [16]. In this paper, the routing model based on reinforcement learning is shown in Fig. 2, which includes agent, environment, action, state and reward [17]. The workflow of the model can be described as follows: the agent perceives the current state of the environment S_t , makes the corresponding action a_t and executes it. The environment returns the reward R_t under the action a_t and reaches the next state. After a certain number of interactions with the environment, the agent can finally learn the optimal strategy.

However, in practical applications, the reinforcement learning method cannot be simply applied to the routing problem. Designing reasonable rewards, actions and states for optimisation goals is the key to solving the problem [18]. Therefore, for the problems mentioned above, this section uses the model in Fig. 2 as the basis to define rewards, states and actions as follows:

2.2.1 Reward

In reinforcement learning, agent learns by trial and error, and gets reward to guide behaviour. Therefore, it is necessary to set reward for agent to find good strategies. In this paper, the goal is to improve the network transmission performance. Although throughput is very important for data centre network, it is more important to ensure that the transmission does not time out. Therefore, this paper needs to maximise the throughput of the network while ensuring that the transmission delay meets the quality of service requirements. Specifically, the setting of the reward function is shown in Formula (2).

$$r_i = \begin{cases} T & T_i < T_{Qos} \\ R & \text{Otherwise} \end{cases} \quad (2)$$

In the above formula, i represents the sequence number of the current scenario, T_{Qos} represents the delay requirement of the current network service quality, T_i represents the average

transmission delay of the completed stream in the current scenario, and R is a constant less than 0, T is the current throughput. When the transmission delay of the completed stream meets the quality of service requirements, the reward is the throughput and guides the agent to learn in the direction of improving the throughput. Otherwise, a constant less than 0 is used to guide the agent to learn in the direction of reducing the delay.

2.2.2 State

The state usually represents the environment. In the routing scenario, the routing strategy is usually determined based on the current network status. In the above, the current network status is represented by a traffic matrix. Therefore, we can set the state to the current traffic matrix of the network, the dimension of which is equal to the number of edge layer nodes.

2.2.3 Action

In the routing scenario, the agent takes action based on the current traffic matrix and transforms them into corresponding routing decision according to certain rules. It can be seen from the above problem description that for each transmission task, the routing decision needs to allocate a reasonable number of sub-paths and subflow allocation ratio. Therefore, the action needs to include elements such as the number of sub-paths and the allocation ratio of sub-flows.

Current routing models based on reinforcement learning mostly take link weights as action, and obtain routing strategies through shortest path algorithms. Although this design has small dimensions and is easy to converge, it can only represent a single-path routing strategy, and cannot represent the complex multipath routing strategy required in this paper. Based on this point, this paper considers the use of destination node-based routing rules as the action. For each node V and target node, a allocation ratio to its neighbor nodes is set. Taking node A as an example, the sub-flow allocation ratio table to the target node is shown in Fig. 3. Among them, the horizontal represents the target node, and the vertical represents the allocation ratio of node A to the neighbor nodes. However, this can only indicate the allocation ratio of sub-streams, and cannot select sub-paths. Therefore, for each component allocation ratio, the model additionally sets a threshold α . When the allocation ratio corresponding to the neighbour node is greater than the threshold, the node is selected as the next hop node. Through this method, the number of sub-paths and the sub-flow

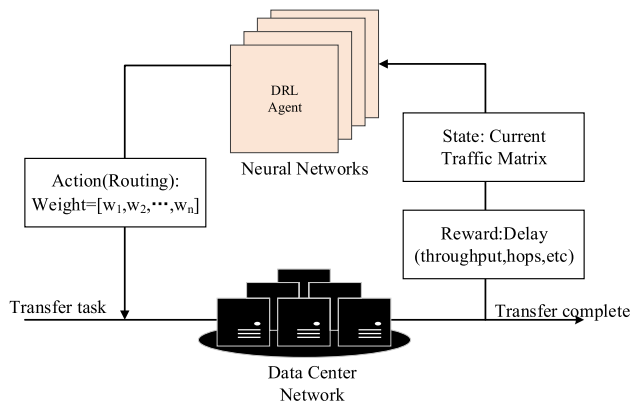
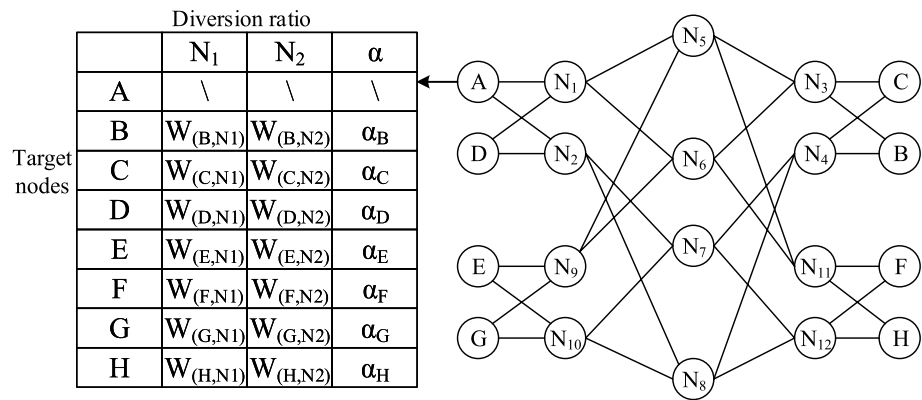


Fig. 2 Reinforcement learning model about routing

Fig. 3 Diversion ratio table of source node A



allocation ratio between the source node–target node pair can be well found.

3 Algorithm Design and Analysis

3.1 Algorithm Implementation

In the multipath routing problem, the role of the reinforcement learning agent is the routing decision maker, which outputs the latest routing path after obtaining network status information. From the time scale, the way that agent takes action is mainly divided into packet-level routing control mode and time-period-level routing control mode [19]. The packet-level routing control method takes action on each packet with finer granularity. The route control method at the time segment level divides the time into multiple time slices and takes actions for each time slice, with a larger granularity.

Considering that the algorithm needs to perform routing control based on the overall status of the network and the large scale of data centre network traffic, we adopt a time-period-level routing control mechanism. The specific content of this mechanism is shown in Fig. 4. In Fig. 4, we use centralised control to place the reinforcement learning agent on the control plane. In addition, in terms of data interaction, there is corresponding data interfaces between the control plane and the data plane. In each time segment, the control plane measures the network status and transmits it to the agent. The agent gives routing instructions according to the network status, and updates the strategy according to the feedback in the next time segment.

In terms of reinforcement learning algorithm selection, although the table-based algorithm shown in Fig. 5 is highly interpretable, it is prone to dimension disaster due to the huge dimensions of the state and action space in the routing problem [20]. Therefore, this paper uses the DDPG algorithm [21] combined with deep learning. The DDPG algorithm uses a neural network instead of a table to represent

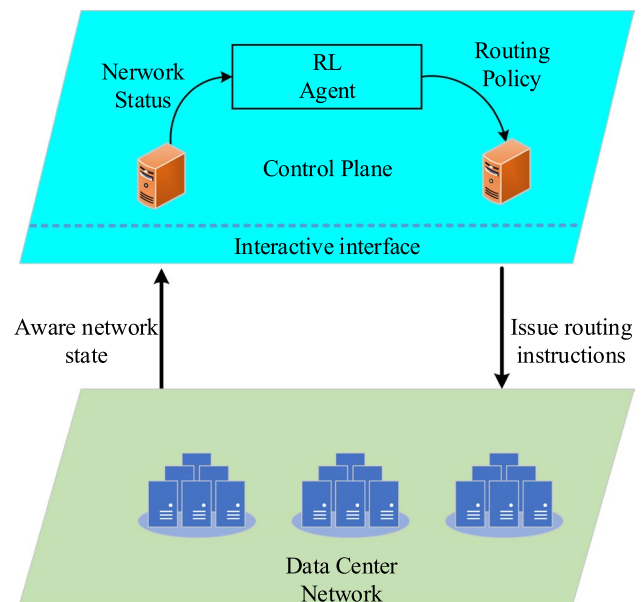


Fig. 4 Route control mechanism based on time segment

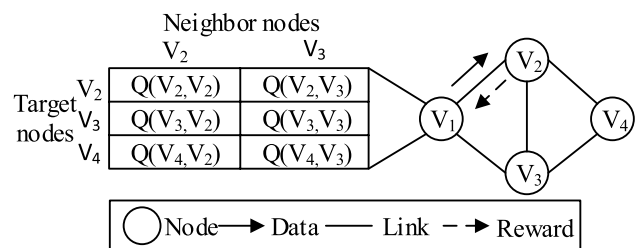


Fig. 5 Table-based routing algorithm structure

the mapping of states to actions in a functional approximation manner. In this way, the problem of dimension disaster can be solved well, and it can fit the routing control mechanism based on time segment well.

On the basis of the above, this paper uses DDPG algorithm to construct agent, and regards the routing problem

as a Markov decision process. Agent, as a routing decision maker, interacts with the network environment with the goal of obtaining the largest cumulative reward. Then when the environment state is S_t , the agent (routing decision maker) perceives the current environment state S_t (traffic matrix), takes action a_t , and converts it into corresponding multipath routing decision execution. The environment returns the reward R_t under the action of a_t to evaluate the quality of the current action (routing instruction). Then algorithm reversely updates the DDPG neural network parameter W to achieve strategy improvement. It goes over and over again. Finally, the DDPG agent learns the optimal routing strategy, and outputs it in the form of neural network weight W . The detailed pseudo code of the algorithm is as follows.

Where Qos_target represents the optimization objective, like latency, for example, when the current network performance does not meet the requirements of the target, it will keep optimising, and when it does, it will stop.

Algorithm 1 SAMP Algorithm for DCN

Input: Traffic matrix M , Graph $G = (V, E, c(v), m, n)$, Qos_target
Output: DDPG neural network parameters W

- 1: while Current transmission performance $< Qos_target$ do
- 2: Get the traffic matrix M ;
- 3: Input M as the *state* into the neural network;
- 4: Obtain the *action* from output vector;
- 5: Convert the *action* into routing instruction and send it to the network;
- 6: Get the *reward* from formula (2) and update the neural network parameters W ;
- 7: end while
- 8: Extract the neural network parameter W and use it as a strategy learned by the Agent

3.2 Deployment Framework

Considering that the algorithm is based on the acquisition and analysis of the real-time state of the network, it needs to accurately perceive the overall state view information of the network during deployment and implementation. On the other hand, it is necessary for the network to have the decision-making ability in logical concentration. It means that the network should have the unified deployment and control ability from the decision center to each related node. Therefore, we select a control mechanism-based SDN architecture, and its structure is shown in Fig. 6.

The specific workflow of the mechanism can be described as follows: in each time segment, the network executes traffic forwarding according to the strategy generated in the previous time segment. Then the mechanism uses the collected current network state as the input of the decision-making module. Agent generates a new control strategy and transforms it into a flow table. Finally, it is sent to the network through SDN for the next time segment execution. By analogy, a closed-loop feedback is formed.

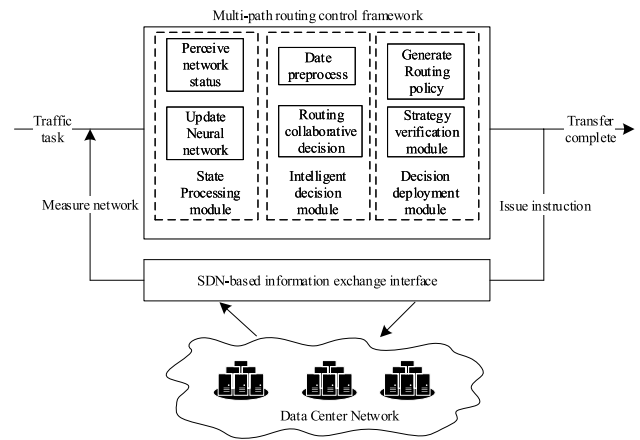


Fig. 6 Routing control mechanism for SAMP

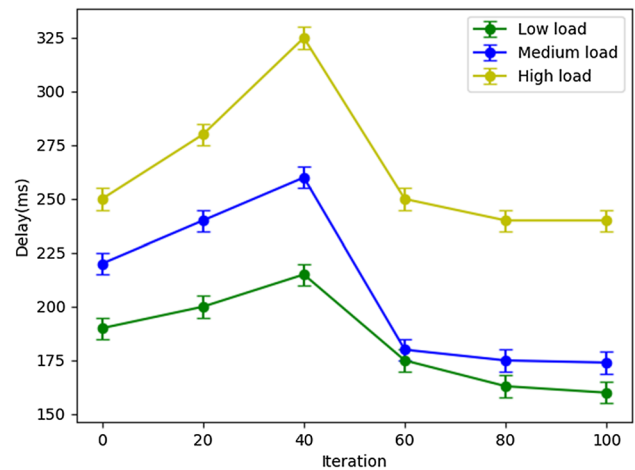


Fig. 7 Schematic diagram of algorithm convergence

3.3 Analysis of Algorithm

In terms of the convergence of the algorithm, this paper conducts experiment on the training of the algorithm under different network loads. The experiment records the change of time delay during training. By changing the transmission delay between each iteration, we can observe the convergence of the algorithm. Because the transmission delay in different network environments fluctuates to a certain extent, the experiment sets a threshold. When the time delay fluctuation is less than the threshold, we believe that the time delay is still in a stable state.

Figure 7 shows the convergence of the algorithm in terms of delay under different network loads. It can be seen from Fig. 7 that under the three network loads of low load, medium load and high load, as the training continues, the transmission delay continues to decrease, and generally stabilises between the iteration schedule of 40–60. At this

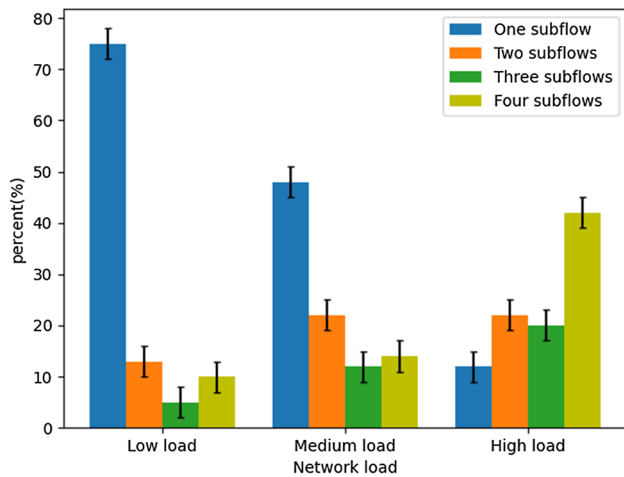


Fig. 8 The adjustment of the algorithm about the sub-path

point, it can be considered that the algorithm finally converges to the optimal value, and the agent has learned a better strategy. Therefore, it can be judged that the algorithm in this paper has good convergence.

Considering the influence of the number of sub-paths on the multipath transmission effect, the algorithm needs to select the number of sub-paths according to the actual network state. Therefore, this paper counts the number of sub-paths selected for transmission tasks between node *A* and node *B* under different network loads.

The result is shown in Fig. 8, where the network state is divided into low load, medium load, and high load. For each network state, Fig. 8 shows the proportion of different sub-paths in the path used by the transmission task in the form of a bar graph. Through comparison, it can be found that more transmission tasks use one or two paths for transmission under low load conditions. As the network load increases and the busyness of network nodes increases, the algorithm adjusts the number of sub-paths. Among them, in the high load state, the algorithm makes 21% and 42% of the transmission tasks use three and four sub-paths for transmission to prevent node congestion. Therefore, it can be seen that under different network conditions, the algorithm in this paper can effectively adjust the number of sub-paths to ensure the transmission effect.

4 Experimental Design and Analysis

4.1 Experimental Setup

To verify the actual performance of the SAMP algorithm in this paper, we design a data centre network multipath transmission control test environment for experiments. The network environment adopts a level-3 fat-tree network

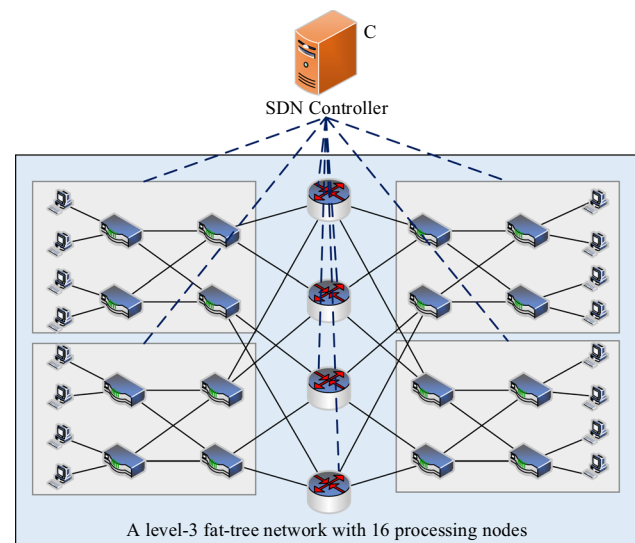


Fig. 9 Experimental network environment

with 16 processing nodes, and the structure is shown in Fig. 9. Among them, different nodes have different buffer capacities. Hosts that support the installation of OpenvSwitch are used to simulate SDN network forwarding equipment. So that it can collect network status information including link load, bandwidth usage and buffer length. Then it can upload the information to the control center, and perform transmission decision issued by the center. Server C installs floodlight as the SDN network controller to obtain overall network status information. At the same time, the routing control mechanism is deployed in server C to make transmission decisions. Then the decision results are deployed to each node.

In terms of network traffic, the experiment generates three different intensities of low, medium and high traffic matrices based on the gravity model. The total amount and distribution of the traffic are different. In addition, the hidden layer of DDPG's actor network consists of three fully connected layers. Each fully connected layer has 30 neurons, and the learning rate is set to 0.0001 to update the network parameters. The critic network of DDPG is composed of three fully connected layers. Each fully connected layer has 50 neurons, and the learning rate is set to 0.0001 to update the network parameters. Both neural networks choose ReLU as the activation function.

The comparison algorithms used in the experiment are as follows: algorithm one uses the traditional OSPF for data transmission; algorithm two uses the equal cost multipath (ECMP) routing algorithm for data transmission; algorithm three uses the weight cost multipath (WCMP) routing algorithm for data transmission. In addition, the experiment assumes that the traffic transmission

is not affected by the link, and the link bandwidth is set to 1000MB. In terms of data statistics, the experiment ignores the background traffic, converts the packet loss rate into the task transmission success rate as a performance indicator. It also sets the statistical unit of delay to *ms* and sets the statistical unit of throughput to *MB*.

4.2 Result Analysis

To study the ability of the SAMP algorithm to calculate the multipath routing and sub-flow allocation ratio in real time according to the network status, we use delay, packet loss rate, and throughput as performance indicators to calculate the difference between the SAMP algorithm and the comparison algorithm under static network load and dynamic network load. Under static network load, we divide the network load into three intensities: low, medium and high. We compare the differences between different algorithms under the same network load. Under dynamic network load, we set the initial network load intensity to low load, and make the network load higher in the iterative process. In this way, we can compare the range of changes in various performance indicators of various algorithms in the face of network load changes. Based on the above content, we design the following three experiments.

4.2.1 Experiments on Transmission Delay Under Different Network Loads

From the perspective of delay, this experiment compares the difference in delay between different algorithms under static and dynamic network loads. Under static network load, the experiment compares the transmission delay difference of different algorithms under different network loads. Under dynamic network load, the experiment compares the

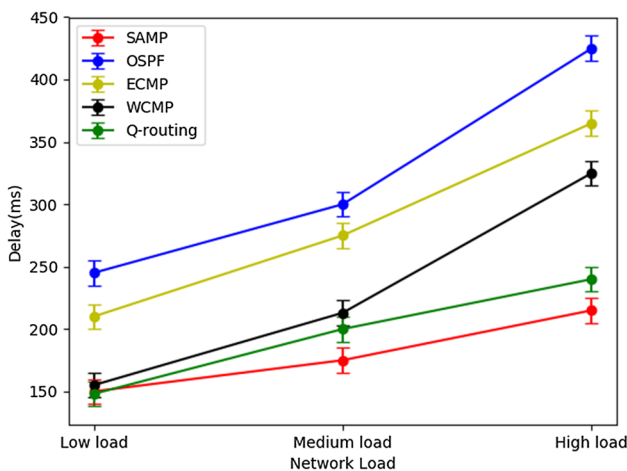


Fig. 10 Transmission delay under static network load

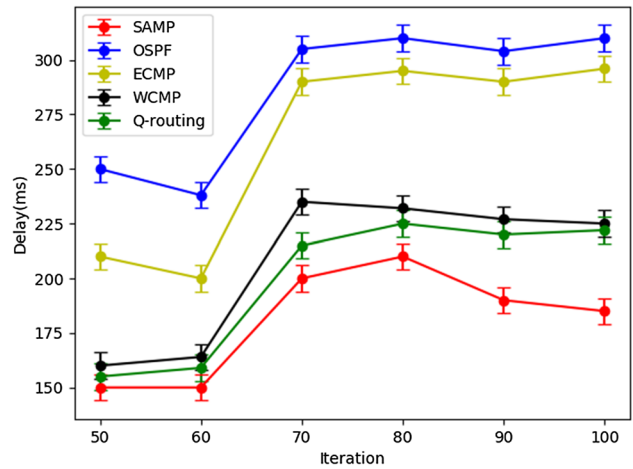


Fig. 11 Transmission delay under dynamic network load

performance of each algorithm before and after the node load changes.

The results of experiment are shown in Figs. 10 and 11. Figure 10 shows the statistics of the transmission delays of different algorithms under different network loads. Figure 11 shows the performance of the various algorithms during the progress of the change of network load.

It can be seen from Fig. 10 that under different network loads, SAMP can provide a transmission delay lower than the other four comparison algorithms, and the gap becomes larger as the network load becomes higher. Especially under high network load, the transmission delay of SAMP is 37.5%, 32.6%, 23.8% and 5.5% less than the comparison algorithms, respectively. This also shows that SAMP can make full use of the advantages of multipath transmission. In Fig. 11, the network load change occurs at the iteration progress of 50–60. At this time, the transmission delay of SAMP has achieved convergence. After the network load change occurred, although the delay increased for a period of time, as the online learning of SAMP progressed, the delay quickly decreased by 12.9% and converged to stability. However, the comparison algorithms cannot effectively adjust the strategy for the real-time network status, which causes the transmission delay to be at a relatively large value.

In a few exceptional cases, like the span size of the network load change greatly, it will affect the effectiveness of the policy. Our approach will implement more iterations to adjust the policy to adapt the new network environment.

4.2.2 Experiments on the Transmission Success Rate Under Different Network Loads

From the perspective of task transmission success rate, the experiment compares the difference in task transmission success rate between different algorithms under static and

dynamic network load. Among them, under the dynamic network load, the experiment counts the difference in the overall task transmission success rate of each algorithm affected by the change of the network load in the iteration.

The results of experiment are shown in Tables 1 and 2. The task transmission success rates of each algorithm under static network load and dynamic network load are respectively counted.

By comparing the task transmission success rate of each algorithm in Table 1 under different network loads, it can be found that each algorithm can maintain a higher success rate under low network load. With the increase of network load, ECMP, WCMP and Q-routing benefit from multipath transmission, and can still carry out task transmission. But their task transmission success rate has decreased to a certain extent. OSPF is limited by single-path transmission. This situation leads to a rapid decline in the success rate of task transmission. Under high load, the task transmission success rate of other algorithms is 40.1%, 17.8%, 6.6% and 2.6% less than SAMP. In addition, Table 2 shows the difference in the task transmission success rate of each algorithm under dynamic network load. In Table 2, the task transmission success rate of other algorithms is 23.1%, 18.3%, 12.1% and 6.2% less than SAMP, respectively. Combining the difference in the task transmission success rate of each algorithm under static and dynamic network load, it can be considered that SAMP can adjust the routing strategy in time to avoid the decrease of task transmission success rate.

4.2.3 Experiments on the Throughput Under Different Network Loads

From the perspective of throughput, the experiment compares the difference in throughput between different methods under different network loads. The result is shown in Fig. 12, which counts the differences in throughput of each algorithm under different static network loads.

By comparing the histogram in Fig. 12, it can be found that as the network load becomes higher, the throughput under each algorithm has a decreasing trend. However, relatively speaking, due to the influence of the task transmission success rate, the throughput of SAMP decreases slightly. And under high load, with the help of good route discovery and sub-flow distribution capabilities, SAMP can still guarantee high throughput. In addition, comparing the changes

Table 1 Transmission success rate in static network load

Network load	SAMP	OSPF	ECMP	WCMP	Q-routing
Low load	99.63%	96.73%	98.85%	99.85%	99.71%
Medium load	97.69%	80.61%	86.63%	89.88%	94.65%
High load	82.19%	42.17%	64.39%	75.58%	79.61%

Table 2 Transmission success rate in static network load

Comparison algorithm	SAMP	OSPF	ECMP	WCMP	Q-routing
Success rate	95.72%	73.11%	77.42%	83.64%	89.57%

in throughput of each algorithm before and after the network load changes, the differences between the algorithms can be found. Even if the network load changes from a low load to a high load, the throughput of SAMP is very small, only a 14% drop. This is due to the online learning capabilities provided by reinforcement learning, which can adjust routing and sub-flow distribution according to the real-time state of the network. In contrast, the comparison algorithms cannot adapt well to the new network environment after the network load changes, and the throughput drops by 57%, 39.3%, 30.2% and 21.9%, which has a greater impact on network transmission.

5 Conclusion

For multipath transmission in data centre network, existing multipath routing methods introduce machine learning, which can adapt to the dynamic changes in data centre networks well. However, these methods lack cooperation in path selection and sub-flow assignment. Therefore, this paper proposes a sub-flow adaptive multipath routing algorithm for data centre network (SAMP). Considering the superiority of machine learning in dynamic environments, the deep reinforcement learning algorithm DDPG is introduced to implement different network state changes quickly, especially in the process of topology, to achieve dynamic migration of the optimised decisions that have been learned. To address the problem of lack of cooperation between path selection and

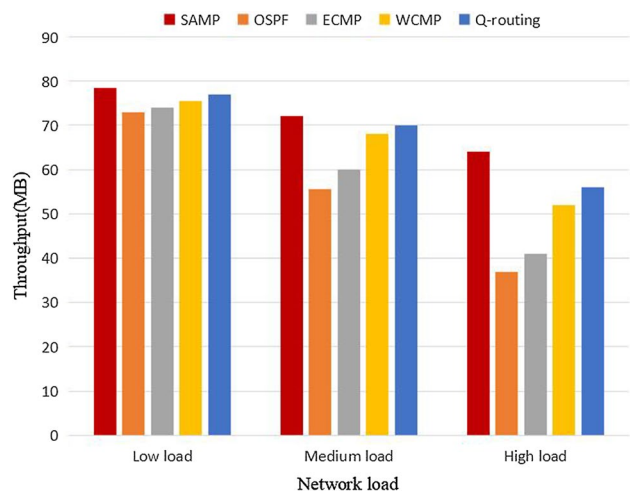


Fig. 12 Throughput under different network load

sub-flow assignment, a sub-flow adaptive multipath routing model is established, it can accomplish collaborative scheduling of path selection and sub-flow assignment according to the real-time state of the network. Experiments show that compared with commonly used routing algorithms, SAMP algorithm can reduce the latency by an average of 31.3%, improve the task transmission success rate by an average of 14.9% and increase the throughput by 37.1%.

On the basis of the above conclusions, we put forward prospects for follow-up research. Considering the influence of model design on the convergence of the algorithm, we hope to optimise the dimensionality of the routing model, and improve the convergence speed and self-adjustment ability of the SAMP algorithm.

Author Contributions YL, YC and LL: conception. YL: experiment and analysis of data. YL, YC, XX and QF: preparation of the manuscript. JC: supervision. All authors contributed to the article and approved the submitted version. All authors have read and agreed to the published version of the manuscript.

Funding This work was financially supported by Primary Research and Development Plan of China (No. 2020YFC2006602), National Natural Science Foundation of China (No. 62072324, No. 61876217, No. 61876121, No. 61772357), University Natural Science Foundation of Jiangsu Province (No. 21KJA520005), Primary Research and Development Plan of Jiangsu Province (No. BE2020026), Natural Science Foundation of Jiangsu Province (No. BK20190942).

Availability of Data and Materials Data and materials can be requested from the corresponding author.

Code Availability Code can be requested from the corresponding author.

Declarations

Conflict of Interest The authors declare no conflict of interest.

Ethics Approval This work accord with ethics approval.

Consent to Participate Approve.

Consent for Publication Approve.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. Fan, F., Y.K.L. Hu, B.: Routing in black box: modularized load balancing for multipath data center networks[c]. In: /IEEE INFOCOM 2019-IEEE Conference on Computer Communications. IEEE (2019)
2. Lu, Y., Xu, M., Zhengzhi X.: FAM: A Flow-Aware Marking and Delay-based TCP algorithm for datacenter networks[J]. *Network. Comput. Appl* **174**, 102912 (2021)
3. Liu, Z., L.M. Zhao, A.: A port-based forwarding load-balancing scheduling approach for cloud datacenter networks[j]. *Cloud. Comput* **10**(1), 1–14 (2021)
4. Besta, M., Domke, J., Schneider, M., et al. High-performance routing with multipathing and path diversity in ethernet and hpc networks[J]. *IEEE Trans. Parallel Distrib. Syst* **32**(4), 943–959 (2020)
5. Cheng, Y.J.X.: Namp: network-aware multipathing in software-defined data center networks[j]. *IEEE/ACM Trans. Network* **28**(2), 846–859 (2020)
6. Awad, M.K., Ahmed, M.H.H., Almutairi, A.F., et al. Machine learning-based multipath routing for software defined networks[J]. *J. Netw. Syst. Manage* **29**(2), 1–30 (2021)
7. Huang, J., Lan, J., Hu, Y., et al. A Segment Routing Based Multipath Flow Transmission Mechanism[J]. *Acta. Electronica. Sin* **46**(6), 1488 (2018)
8. Li, W., Liu, J., Wang, S., et al. Survey on traffic management in data center network: from link layer to application layer[J]. *IEEE Access* **9**, 38427–38456 (2021)
9. Cai, Y., Wang, C.: Software defined data center network hybrid routing mechanism[j]. *J. Commun.* **37**, 44–52 (2016)
10. Maksić, N.: Two-phase load balancing for data center networks using openflow[c]. 2017 25th telecommunication forum (telfor). In: IEEE, pp. 1–4 (2017)
11. Cheng, C.: Research on Reinforcement Learning Based Routing Planning Algorithm in sdn [d][j]. Zhejiang Business University (2018)
12. Rischke, J., Salah, H., Sossalla, P.: Qr-sdn: towards reinforcement learning states, actions, and rewards for direct flow routing in software-defined networks[j]. *IEEE Access* **8**, 174773–174791 (2020)
13. Xu, X., Gu, L., Chen, J., et al. An intelligent multipath routing and subflow distribution cooperative algorithm[J]. *Comput. Eng* **047**(009), 136–144 (2021)
14. Fu, Q., Sun, E., Meng, K., et al. Deep Q-learning for routing schemes in SDN-based data center networks[J]. *IEEE Access* **8**, 103491–103499 (2020)
15. Yang, Y., Yang, J.-H., Wen, H.-S.: Routing algorithm design based on timeslot of transmission for data centers[J]. *J. Software* **29**(8), 2485–2500 (2018)
16. Li, Y.: Deep reinforcement learning: an overview [J]. arXiv preprint [arXiv:1701.07274](https://arxiv.org/abs/1701.07274) (2017)
17. Liu, W.: Intelligent routing based on deep reinforcement learning in software-defined data-center networks[c]. In: 2019 IEEE Symposium on Computers and Communications (ISCC). IEEE, pp. 1–6 (2019)
18. Pinyoanuntapong, P., W.P. Lee, M.: Delay-optimal traffic engineering through multi-agent reinforcement learning[c]. In: IEEE INFOCOM 2019-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). IEEE, pp. 435–442 (2019)
19. Xu, Q., Zhang, Y., Wu, K., et al. Evaluating and boosting reinforcement learning for intra-domain routing [C]. In: 2019 IEEE 16th International conference on mobile Ad Hoc and sensor systems (MASS). IEEE, pp. 265–273 (2019)

20. Liu, C.Y., Xu, M., Geng, N., et al. A survey on machine learning based routing algorithms[J]. *J. Comput. Res. Dev* **57**(4), 671–687 (2020)
21. Luong, N. C, Hoang, D. T., Gong, S., et al. Applications of deep reinforcement learning in communications and networking: a survey[J]. *IEEE Commun. Surv. Tutorials* **21**(4), 3133–3174 (2019)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.