SDN routing optimization based on improved Reinforcement learning

Xiaoye Lu
Guangxi Key Laboratory of cloud computing and complex systems
Guilin University of Electronic Technology
China Guilin
1834208304@qq.com

Junyan Chen†
Guangxi Key Laboratory of cloud computing and complex systems
Guilin University of Electronic Technology
China Guilin
56546632@qq.com

Liaochuo Lu
Guangxi Key Laboratory of cloud computing and complex systems
Guilin University of Electronic Technology
China Guilin
530144946@qq.com

Xuefeng Huang
Guangxi Key Laboratory of cloud computing and complex systems
Guilin University of Electronic Technology
China Guilin
2015732115@qq.com

Xiantao Lu
Guangxi Key Laboratory of cloud computing and complex systems
Guilin University of Electronic Technology
China Guilin
528246923@qq.com

ABSTRACT

In recent years, the application of new network architecture SDN combined with reinforcement learning in the field of artificial intelligence is more and more extensive, considering the diversity and complexity of traffic transmission. This paper proposes a DDPG-EREP(Enhanced and renewable experience pool) algorithm that dynamically plans the experience pool capacity and sampling size according to the current iteration number, and can update the experience in time, and applies it to SDN routing optimization. In addition, good results have been achieved through experiments, including better training speed and better routing optimization of SDN.

CCS CONCEPTS

•Networks~Network algorithms~Control path algorithms~Network control algorithms

KEYWORDS

SDN；Reinforcement Learning；Routing optimization;DDPG

CIAT 2020, December 4–6, 2020, Guangzhou, China

ACM ISBN 978-1-4503-8782-8/20/10…$15.00
https://doi.org/10.1145/3444370.3444563

1 Introduction

In recent years, with the continuous expansion of network scale, network traffic also presents a huge growth, and the need for network control is also becoming more sophisticated. Nick McKeown, Tom Anderson and others [1] put forward the concept of OpenFlow for the first time, marking the birth of SDN(Software Defined Network). However, the traditional network has obvious disadvantages, such as slow convergence speed and difficulty in dealing with network congestion. Therefore, efficient optimization algorithm for network routing through SDN controller is the key factor to ensure network service and promote the development of SDN. For example, Ye Ye [2] proposed SDN load balancing method based on K-Dijkstra algorithm. The average delay, average packet loss rate, and hops are used as edge weights. The experiment determined the shortest path of the directed graph and deleted one of the edges. On this basis, delete one of the edges, and select an alternative path based on K alternative paths determine the shortest path of the directed graph and delete one of the edges. On this basis, delete one of the edges, and select an alternative path based on K alternative paths. Song Wenwen, Wang Jun et al, [3] proposed a dynamic load balancing mechanism based on particle swarm optimization for elephant flow in the data. Based on particle swarm optimization algorithm and based on the current network link resource state and other information, the optimal transmission path is calculated for elephant flow by combining SDN with technical advantages such as global network topology information visualization and minimizing the maximum link utilization rate. Dong Shuai, Zhang Anlin et al, [4] proposed the topology discovery optimization of SDN based on OpenFlow. The improved mechanism based on

OFDP accelerates the generation of network topology and reduces the load on the controller.

In recent years, machine learning has excelled in large-scale data processing, classification and intelligent decision making. Many researchers have begun to attempt to solve routing optimization problems through machine learning. Che Xiangbei, Kang Wenqian et al, [5] proposed a SDN routing optimization algorithm based on reinforcement learning. This algorithm uses PPO deep reinforcement learning mechanism to optimize the routing of SDN, thus realizing real-time intelligent control and management of SDN. However, it is difficult to obtain a large number of training samples in the face of large-scale complex networks. Giorgio Stampa, Marta Arias et al, [6] designed an agent that optimizes routing, which can reduce network delay by automatically adapting to current traffic conditions and proposing configurations.

Timothy P. Lillicrap, Jonathan J. Hunt et al, [7] proposed for the first time an actor-critic, model-free DDPG(Deep Deterministic Policy Gradient) algorithm based on a deterministic policy gradient and running in a continuous action space. Luo Ying, Qin Wenhu et al, [8] proposed DDPG improved algorithm DDPG-CBF combined with CBF controller. CBF controller is used for safety compensation control and policy exploration guidance. Su Shihui, Lei Yong et al, [9] used the MULTI-agent DDPG algorithm and constructed a global optimal collaborative control system, which achieved good results. Liu Yandong [10] proposed the DDPG-VCEP(Variable Capacity Experience Pool) algorithm based on the learning curve theory, and improved the learning rate of the algorithm to a large extent by dynamically changing the capacity of the experience pool and sampling size. However, the growth rate of the experience pool capacity of the algorithm is not equal in the early stage and the late stage, which results in the loss of the early stage state and the obsolescence of the late stage state. In this paper, DDPG-EREP is proposed to increase the capacity of the experience pool gently through function changes. And the algorithm is applied to SDN. In SDN, the ratio of the number of bytes passed per unit time to the reward in the previous state is used as a reward. This algorithm is used to change the packet queue in the switch to optimize the link load. The basic ideas and core concepts of the algorithm are explained in detail below.

2 Related Technology

2.1 SDN

SDN is a new type of network architecture. Its design concept is to separate the control plane of the network from the data forwarding plane, so as to realize the gradual control of rotating hardware through the software platform in the centralized controller, and realize flexible network resources. Deployment on demand.In the SDN network, the network equipment is only responsible for pure data forwarding, and general hardware can be used; The operating system originally responsible for control will be refined into an independent network operating system, which is responsible for adapting to different service characteristics, and the communication between the network operating system and service characteristics and hardware devices can be realized through programming.

Compared with traditional networks, SDN has three basic characteristics:

1) Separation of control and forwarding.

The control plane of the network element is on the controller, which is responsible for protocol calculation and generating flow tables; while the forwarding plane is only on the network equipment

2) The open interface between the control plane and the forw

SDN provides an open programmable interface for the control planearding plane.

3) Logical centralized control.

The logically centralized control plane can control multiple forwarding plane devices, that is, control the entire physical network, so that a global network status view can be obtained, and the network can be optimized and controlled according to the global network status view.

SDN architecture diagram is shown in Figure 1

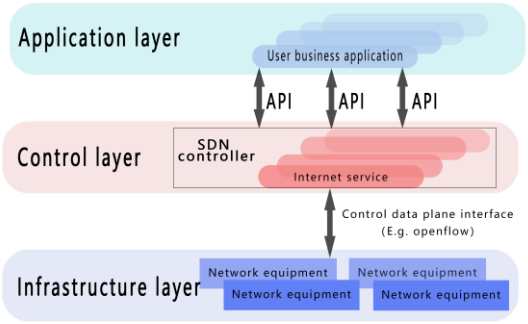


Figure 1: SDN architecture diagram

2.2 Reinforcement Learning

Reinforcement learning is the third machine learning method along with supervised learning and unsupervised learning. It USES an agent to constantly explore the surrounding environment, and each decision it makes yields a corresponding reward or punishment. Then the agent gets feedback signal according to the reward and punishment value, so as to optimize the agent's decision. Reinforcement learning, as unsupervised learning, does not require the same labeling of data as supervised learning.

Reinforcement learning has three basic elements:

(1) The state of the environment S, the state of the environment at time T is a certain state in its environmental state set.

(2) The action taken by an individual At time T is A certain action in its action set.

(3) The reward R of the environment, the reward Rt+1 corresponding to the actions taken by individuals in the state St At time T will be obtained At time T +1.

Reinforcement learning model mainly includes the following components: system environment (Env), Agent (Agent), reward function (R), and selection strategy (Action). As shown in **Figure 2**. The agent selects a strategy A at time T(t), and interact with the

environment through this policy. The agent observes the next state and gets a specific reward or penalty. The reward or penalty may be positive or negative. The agent keeps trying to interact with the environment. After a limited number of attempts, the agent can finally make the optimal strategy in the current environment.

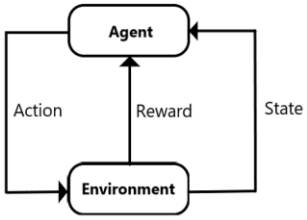
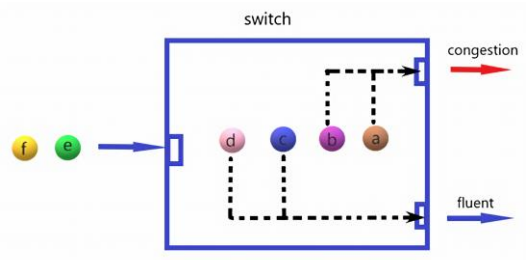


Figure 2: Framework of Reinforcement Learning

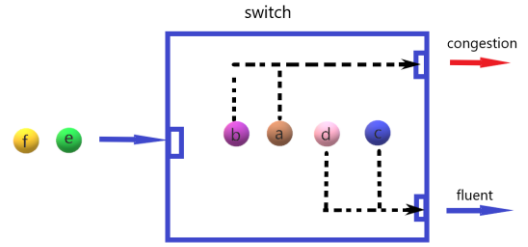
2.3 The effect of reinforcement learning in SDN

When the current SDN link is large, the link will cause congestion, and it is difficult for the SDN controller to operate the flow table in the switch according to the current traffic conditions. When there is a data packet forwarding queue on the data plane switch, the data packets at the front end of the queue are stuck due to the congestion of the forwarding port and the forwarding route of the data packet, so that the data packets at the back end of the queue cannot be forwarded smoothly, and the congestion is aggravated.

Reinforcement learning learns the network status in SDN to adjust the packet queue of the data plane switch, so that the data packets at the head of the queue are adjusted to the end of the queue, and the packets with a smooth forwarding path are forwarded first. Thus, the utilization rate of the data link is increased, and the problem of traffic load balancing in SDN is solved. Traditional algorithms can optimize the forwarding strategy of the network to a certain extent, while the method proposed in this paper can adjust the forwarding queue of data packets in the switch. Figure 3 shows the basic principle of optimized SDN proposed in this article. The path (next hop or a few hops in the future) where packets a and b in Figure 3（a) will be forwarded is congested。Data packets a and b are at the head of the data packet queue. If data packet forwarding is still carried out in this way, it will cause the retention of data packets a and b, resulting in data packets c and d not being forwarded smoothly. The limited storage space of the cache queue in the switch will cause the discarding of packets e and f, which will lead to an avalanche crash of SDN. Therefore, if the queue order of the data packets in the switch can be updated according to the status obtained by the SDN global flow entry (forwarding strategy), as shown in Figure 3(b), the data packets c and d are adjusted to the head of the data packet forwarding queue and forwarded first. When c and d are forwarded smoothly, the links of a and b are likely to enter a non-congested state. As a result, the entire SDN will become more and more healthy and efficient. This is due to the efficient learning of the agent **Agent** of reinforcement learning.



(a): Switches that have not adjusted the packet forwarding queue



(b): Switches with packet forwarding queue adjustment

Figure 3: the basic principle of optimized SDN proposed

2.4 Network Framework

In the architecture of SDN, it is divided into data plane, control layer and application layer. The basic environment we run includes openflow switches, hosts, controllers, and application layer-based reinforcement learning agents. The data plane is responsible for receiving the flow table issued by the controller of the control layer to forward the data packet, and responding when the controller needs to adjust the queue of the data packet. The controller needs to capture the network status (bandwidth occupation, traffic, and delay) of the data plane, and deliver the flow table policy to the data plane switch. The controller upwards needs to provide the current global flow entry and network status, and the network status is used as a reward for data link action processing by reinforcement learning. The controller also needs to receive the actions made by the agent and adjust the sequence of the data packet forwarding queue of the data plane switch. Therefore, the optimization of the SDN network by the agent in this article does not depend on the controller's packet forwarding strategy. The Agent agent we deployed interacts with the SDN network model environment at the application layer of the SDN network. The SDN controller collects the flow entry information of the global switch as the state S, and then transmits it to the agent agent for training through the northbound interface. According to the current state S, select action A to queue the flow entries in the switch through the northbound interface. Finally, the number of bytes transmitted by the flow entry is obtained through the northbound interface, and then the ratio of the number of bytes passed in a unit time to the reward of the previous state is used as the reward R. The network framework of this article is shown in Figure 4.

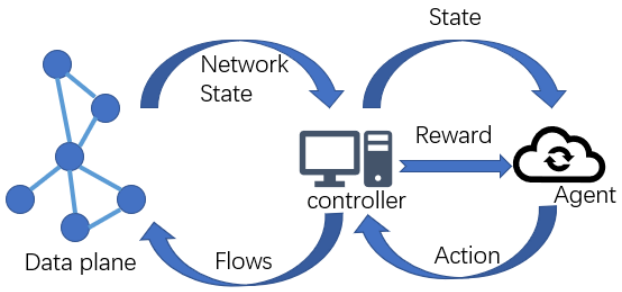


Figure 4: SDN-based network framework diagram

3 DDPG-EREP algorithm

The traditional DDPG algorithm applied to SDN training process has the problem of too slow learning rate, and the fixed experience pool capacity and sampling size also limit the diversity of samples. Therefore, this article uses the DDPG-EREP algorithm to dynamically plan the experience pool capacity and sample size and smooth the experience pool capacity according to the current iteration number, so that the growth rate of the previous experience pool is accelerated, and the previous experience is solved by reducing the experience pool capacity in the later Issues such as obsolescence. An empirical pool capacity variation function is constructed according to the algorithm:

$$D = D(k(i)^{-\alpha})^{-\frac{\log\gamma}{\log 2}} \quad (1)$$

Sample size change function:

$$N = N(i)^{-\frac{\log\gamma}{\log 2}} \quad (2)$$

Where k is a constant, α and γ are learning coefficients, and i is the current iteration number.

State S: Use the flow entry being transmitted in the switch in the network simulation.

Action A: The order of the flow entry queues in the flow table in the controller is given through the northbound interface, and then the flow table is sent to the switch to change the flow forwarding sequence.

Reward R: The ratio of the number of bytes passed in a unit time to the reward of the previous state DDPG-EREP algorithm description

(1) Actor current network, Actor target network, Critic current network, Critic target network, The parameters are θ, θ′, w, w′, attenuation factor γ, soft update coefficient τ, experience pool capacity D, and batch gradient descent sample number N. Current iteration number K . The maximum number of iterations T. Initialize the Agent. Initialize the network simulation and controller.
(2) Whether the maximum number of iterations is reached T. Is the next step, or the end of the program.
(3) Initialize the environment and get the first state S.
(4) In the current network of the Actor, the action A = θ(S) is obtained based on the state S.
(5) Perform action A in the network simulation. Get the new state S'and reward R, whether to terminate the state done.
(6) Store the five-tuple (S,A,R,S′,done) into the experience playback set D.
(7) S=S'.
(8) Sampling m samples {Sj,Aj,Rj,S′j,done },j=1,2,,,m,from the experience playback set D, calculate the current target Q value yj:

$$y_j = \begin{cases} R_j & done\ true \\ R_j + \gamma Q'(S'_j, \pi\theta'(S'_j), w') & done false \end{cases} \quad (3)$$

(9) Use the mean square error loss function $\frac{1}{m}\sum_1^m(y$ j-Q(Sj,Aj,w))2 to update all the parameters w of the current Critic network through the gradient back propagation of the neural network.
(10) Use $J(\theta)=-\frac{1}{m}\sum_{j=1}^{m} Q(si,ai,\theta)$ to update all the parameters θ of the current Actor network through the gradient back propagation of the neural network.
(11) Update the Critic target network and Actor target network parameters:

$$w' \leftarrow \tau w + (1-\tau)w' \quad (4)$$

$$\theta' \leftarrow \tau\theta + (1-\tau)\theta' \quad (5)$$

(12) Dynamically change the fixed experience pool capacity and sample size of the traditional DDPG according to the number of training i: (1) (2)
(13) If S′ is in the terminal state, the current iteration is completed, otherwise go to step 2.

The flowchart is shown in Figure 5

4 Experiment

In this experiment, under the Ubuntu18. 04 system, the DDPG algorithm and DDPG-EREP algorithm were respectively tested using RYU controller and Mininet simulation tools. This paper conducts 50algorithm training experiments. The initial capacity of the design experience pool is 100, the sample size is 5, and the reward discount factor α = 0. 8, γ = 0. 9.

The topology is shown in the Figure 6.

The red curve in Figure 7 is the DDPG-EREP algorithm to get the reward convergence curve, that is, the traffic size of the link between the switches in the SDN per unit time. Therefore, it can be concluded that the data packet forwarding efficiency and routing load balance of the data plane using the DDPG-EREP algorithm have been greatly improved. Compared with the traditional DDPG algorithm and the unmodified DDPG-VCEP algorithm, the DDPG-EREP algorithm proposed in this paper can converge faster and has a better effect on load balancing. This is

due to the increase in the experience pool capacity of the DDPG-EREP algorithm and the change in the sample size. In the DDPG-VCEP algorithm, the rapid growth of the experience pool in the later period causes the experience to be updated not in time, so the convergence effect is not ideal.

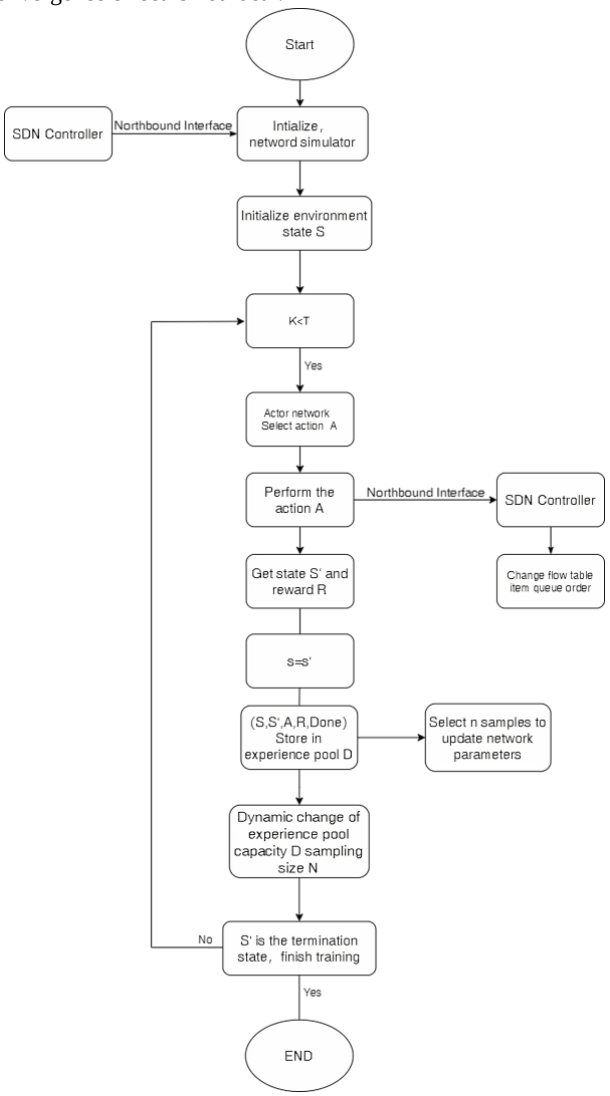


Figure 5: Algorithm flowchart

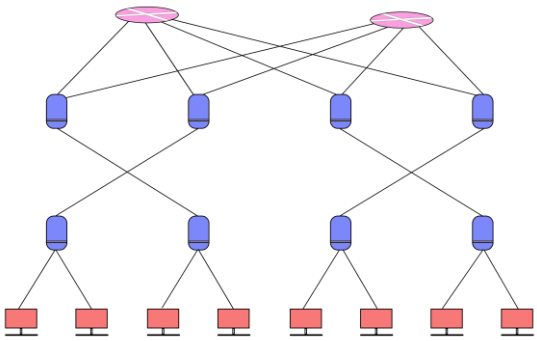


Figure 6: Topology

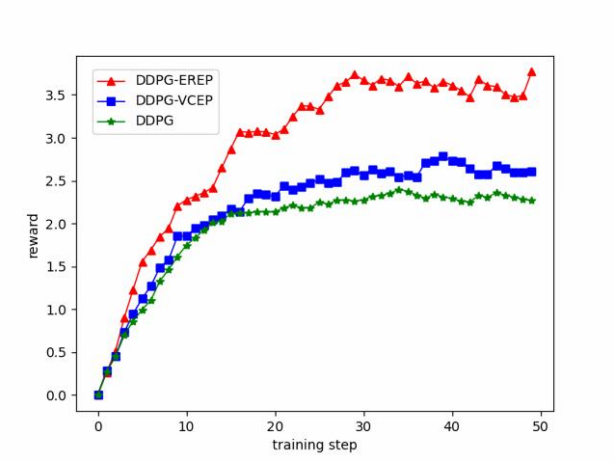


Figure 7: Algorithm flowchart

5 Summary

In this paper, the algorithm is optimized based on SDN architecture and DDPG-EREP. Firstly, the state is obtained on SDN by an agent, and the agent selects actions to interact with the environment. In this paper, the ratio of the number of bytes passed per unit time to the reward in the previous state is used as the reward, and the experience pool capacity and sample size can be dynamically changed in real time according to the number of iterations. After optimization training, satisfactory results can be achieved in SDN routing optimization. Admittedly, there are still some shortcomings in this paper, such as the slow reading of information on complex topologies, which will be further studied and improved in the future.

ACKNOWLEDGMENTS

This work is supported by Project to Improve the scientific research basic ability of Middle-aged and Young Teachers（No.2020KY05033)

REFERENCES

[1] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. 2008. OpenFlow: enabling innovation in campus networks. SIGCOMM Comput. Commun. Rev. 38, 2 (April 2008), 69–74.

[2]YE Ye. Research on SDN Load balancing Strategy based on K-Dijkstra Algorithm [J]. Electronic Technology & Software Engineering,2018(24):14-15.

[3] SONG Wenwen，WANG Jun，DU Ye，XU Jingming，LI Chengxing. Load balancing technology for data center based on particle swarm optimization [J]. Journal of Nanjing University of Posts and Telecommunications(Natural Science Edition),2019,39(05):81-88.

[4] DONG Shuai ZHANG An-lin HUANG Dao-ying WANG Xuan-li LIU Jiang-hao. Optimization of Link Layer Topology Discovery in SDN Based on OpenFlow [J]. Fire Control & Command Control,2020,45(08):148-153.

[5] CHE Xiangbei KANG Wenqian OUYANG Yuhong YANG Kehan Ll Jian. SDN Routing Optimization Algorithm Based on Reinforcement Learning [J/OL]. Computer Engineering and Applications:1-7[2020-11-18]

[6]Stampa G, Arias M, Sánchez-Charles D, et al. A deep-reinforcement learning approach for software-defined networking routing optimization[J]. arXiv preprint arXiv:1709. 07080, 2017.

[7] Lillicrap T P, Hunt J J, Pritzel A, et al. Continuous control with deep reinforcement learning[J]. arXiv preprint arXiv:1509. 02971, 2015.

[8] LUO YingQIN Wen-hu ZHAl Jin-feng. Research on Low-Speed Car-Following Policy Decision Based on lmproved [J]. Measurement & Control Technology,2019,38(09):19-23.
[9] su shihui LEl Yong Ll Yongkai ZHU Yingwei. Study on Short-to-Medium-Term Photovoltaic Power Generation Forecasting Model Based on lmproved Deep Deterministic Policy Gradient [J]. Semiconductor Optoelectronics,2020,41(05):717-723.
[10] Yandong Liu, Wenzhi Zhang, et al. Path planning based on improved De ep Deterministic Policy Gradient algorithm [C]. Information Technology, Networking, Electronic & Automation Control Conference, IEEE. 2019.