ATLANTIS PRESS
Advances in Engineering Research (AER), volume 131
3rd Annual International Conference on Electronics, Electrical Engineering and Information Science (EEEIS 2017)

Research on Programmable Data Plane Load Balancing based on Multipath Selection

Li-Qiong HE[1,a], Ling OU[1,b,*]

[1]College of Computer and Information Science, Southwest University, ChongQing , China

[a]565409058@qq.com, [b]ouling@swu.edu.cn

*Corresponding author

Keywords: Programmable data plane, Load balancing, P4 language.

Abstract. The switching equipment on traditional networks have limited capabilities and the exchange of the switching equipment is often associated with long time frame and high development cost. SDN deployment controller under the flow table rules also exists to handle message delay. In this paper, we propose a scheme based on multipath selection of programmable data plane load balancing. We use the P4 language to directly write the data plane flow control multipath program, and then use the improved equivalent multipath routing path matching and port selection to achieve the path efficient allocation and traffic load balancing scheduling. Finally, the NS2 simulation results show that, compared with the traditional equivalent multipath, the improved multipath selection scheme can lower the end-to-end delay, reduce the packet loss rate and improve the throughput.

Introduction

In recent years, with the introduction of "Internet +" thinking, as well as mobile Internet of things, cloud computing technology is booming and there is more and more service and data transmission in the network. This has put increasing demand the size, security and reliability of computer networks. To address this issue, the Software Defined Network (SDN) [1] proposes the idea of separating the control plane from the data plane. The control plane controls the data forwarding plane centrally through OpenFlow protocol, and makes transparent the upper layer application. SDN allows the computer network to have flexible central control of the data plane switch and provides programmability for the control plane as well. Nonetheless, the control plane functionality achieved by software programming is also available on traditional high-level switches and routers. The supplier hardwires these functions in the system and hardware which cannot be easily changed, so the third party will find it difficult to intervene in custom or secondary development.

Although SDN provides flexible programmable ability, the data plane does not have the ability. The data plane is still forwarded through OpenFlow protocol. If OpenFlow needs to support the new forwarding scheme, then the OpenFlow specification will be expanded and modified, which requires the expansion of memory and increases design cost. With the increase in data center load, the number of switches needed will also increase. The traditional equipment can’t be extended, and the suppliers need to redesign the previous hardware. This will lead to many problems such as high update cost and lengthy design cycles, which limits the rapid development of the network. In the deployment of SDN, we will find the situation where the data center network node and new traffic flow continue to increase. This limits the network processing ability of SDN data center. The SDN data center controls the message processing delay, and cannot effectively use the available network capacity to achieve load balancing.

Based on the above problems, this paper aims to make a data plane with programmable ability, avoiding the traditional mode of fixed function restrictions. Packet parsing and forwarding process can also be programmed to achieve scalability of the data plane to improve the overall network performance. We design the data flow processing process using the P4 programming language. Use it instead of requesting the control plane to select the best path for each new traffic, on the least congested path in the network. This can achieve the effective allocation of traffic, which is conducive to the scheduling of traffic load balancing. In addition, the network operators can modify the program according to their own needs, and switch the switch function. They don't need to bind a specific protocol format with the supplier, making the data plane forwarding more flexible and scalable.

Related work

SDN as one of the top ten technologies for future development, has attracted wide attention from academia and industry. As a new technology, it is still in the period of development. The researches mainly focus on the design of network architecture, reliability, flexibility, security and load balancing. This paper mainly introduces the research of programmable data plane and load balancing.

Research on Programmable Data Plane

In traditional network switching equipment, commonly used switch chip technology CPU, ASIC (Application-Specific Integrated Circuit), FPGA (Field Programmable Gate Array), NP (Network Processor) and so on [2]. But the hardware switch chips data processing performance are limited, this made it difficult to achieve large-scale network forwarding. Needs a flexible, scalable technology to achieve data plane forwarding, so put forward the thinking of a programmable data plane.

In recent years, research on programmable is increasingly intense, especially in the SDN was put forward. The future network researchers are increasingly interested in SDN control plane and data plane, flexibility, scalability and other aspects, and the programmable data plane has also become one of the main concerns of SDN. For example, in [3], a general flow instruction set is used to realize the specific protocol configuration forwarding and data link state storage, and the programmability of the data plane is realized, and the network cost is reduced. In [4], the small data packet processing program is embedded into the data plane to achieve the effective scheduling of traffic, through the programmable data plane, to achieve more packets detection and processing. In addition, a new state data plane architecture (SDPA) for SDN data plane is proposed in [5]. The switch status information is managed by designing protocol processing units, forwarding processors, new instruction sets, and status tables. There are also such as Intel's FlexPipe [6], Cavium's Xpliant [7] and Cisco's Doppler [8] and the other business, using similar programmable pipelines to improve data plane performance. In [9], they proposed a idea of programmable packet scheduler, in the switch to prepare the scheduling algorithm to determine the order of packet scheduling and queue priority.It achieved the programmable hardware, but the increase design cost. In [10] introduced a high-level language programming data plane algorithm, and these programs can be compiled on the emerging switch chip to run at low speed microcode. The cost saved can be achieved by designing continuous packet processing and isolation modules to achieve line rate programmability of state algorithms.

Existing research has focused on hardware deployment, data plane monitoring, traffic scheduling and so on, and did not map the program to a variety of target hardware switches, curing hardware processing methods lead to data plane function not scalable, lack of flexibility, so it is important to design a flexible programmable data plane.

Research on Load Balancing

In order to optimize the network performance and ensure the quality of service requirements, there has been a great deal of research based on SDN load balancing. The most famous is Google in 2013 in Sigcomm published an article on the Google B4 network [11], describes the SDN based on data center between the flow of engineering solutions and deployment to achieve. Research on SDN load balancing is mainly based on the algorithm to calculate the different path allocation scheme to meet the requirements of network load balancing. In [12], with the transmission response time and the transmission rate to evaluate through the SDN technology, in the POX compared controller using the polling scheduling and random scheduling on the network load balancing. In [13], they proposed a scheme to solve the load balancing using SDN global view of the network, and combine the BP artificial neural network model to predict the load of different paths.What's more, select a minimum load as the data flow transmission path. This scheme can optimize the path load, reduce network delay, but not for multi-topology structure. In [14], a load balancing mechanism based on OpenFlow is proposed. The flow of the entire OpenFlow network is determined by the multipath forwarding of the K-shortest path algorithm, which can effectively relieve network congestion and reduce the transmission delay, so as to improve the overall performance of the network. But the implementation process is more complex, not easy to traffic statistics. In [15], an adaptive task scheduling strategy based on dynamic load adjustment is proposed. The strategy has the characteristics of high efficiency and load balancing, but the algorithm is more complex and the execution time is longer.

At present, the researches on SDN mainly focus on the control plane. Through the deployment of the controller, it can improve the flexibility and scalability. The load balancing of SDN is achieved by improving the path algorithm. These schemes can improve the performance, but the investment cost is too large. Moreover, by controlling the plane under the surface of the way, the switch will wait for too long and lead to delay. So we need to adjust it to meet the needs of the fast-growing network.

Research on Programmable Data Plane

P4 Language

P4 (Programming Protocol-Independent Packet Processors, P4) [16] is a programmable, protocol-independent packet processing language. It uses the form of pipeline, with protocol independence, target independence and reconfigurability of the three characteristics. The packets enter an ingress pipeline containing a series of match-action tables, and then switched to the output ports. Any network data plane protocol and packet processing behavior can be achieved through a custom packet parser, a match-action table matching process, and a flow control program. The match-action tables specifies the fields to be matched and the matching method. The flow control program specifies the application order of the match-action tables. The use of P4 language exchange processing program, which allows programmers to modify the program according to their own needs, achieves new or existing data processing protocols. In addition, there is no need to redesign of new hardware, can effectively save the cost, shorten the design cycle, improve the overall the network performance. The OpenFlow protocol is compared with the P4 language shown in table 1.

Table 1. OpenFlow protocols with P4 language comparison

	The same point	The different points
OpenFlow protocol	Match-action table	Complex specifications, functions, the new agreement development cost is high, long, table extension problems, lack of flexibility
P4 Language		Modify, compile, extensible, without investment in the design of new hardware

The process of using P4 language to process data packets is shown in Figure 1.

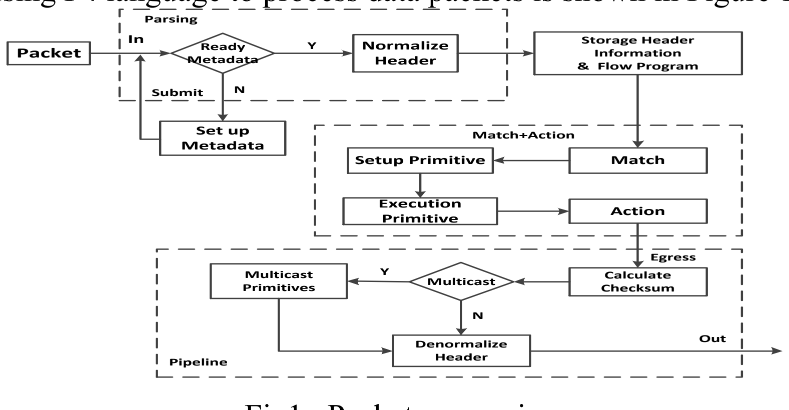

Fig1. Packet processing

Ideas and Processes

Improved ECMP Design Ideas

The traditional Equal-Cost Multipath Routing (ECMP) [17] by CRC16 algorithm to calculate a simple five-tuple (src IP, src port, dest IP, dest port, and protocol) hash value for the path selection. This means that multiple packets belonging to the same service flow may produce the same hash value, which is assigned to the same path, the uneven traffics lead to congestion. In this paper, we propose an improved idea of equivalent multipath routing. When the new data packet arrives, the switch pipeline locks the data packet. After that gets the current path utilization information and adds a certain proportion parameters to the path according to the utilization information. The last, we use the five-tuple hash value and flow ID for path matching and port selection, enabling it to dynamically select, to achieve load balancing, thereby improving the overall performance of the network.

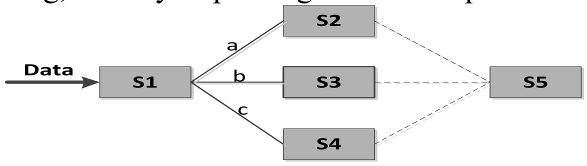

Fig2. Path selection

Assuming the path selection shown in Figure 2. When the data packet to S1, you need to select the next path, the traditional ECMP through the hash value to choose, may lead to always select S1-> S3 this road, so there will be crowding. In order to avoid the situation each time you select the same path, for each path to add a proportional parameter, by calculating the current path utilization rate U_i :

$$U_i = \frac{C_i - L_i}{C_i} \tag{1}$$

Where C_i is the link bandwidth of path i , L_i represents the remaining available bandwidth of path i , and then calculates the path parameter P_i according to the current link utilization rate :

$$P_i = \frac{1-U_i}{\sum_{i=1}^{n} U_i} \tag{2}$$

The larger the link utilization rate U_i is, then the smaller the allocated proportional parameter P_i , the lower the probability of being selected, thus avoiding the congestion of the link. For the convenience of the research, it is assumed that the path utilization is more than 70% when path congestion. When the path was found to be crowded, this path as the current least available path, needs to select the remaining available path. Figure 3 shows the ECMP flow chart.

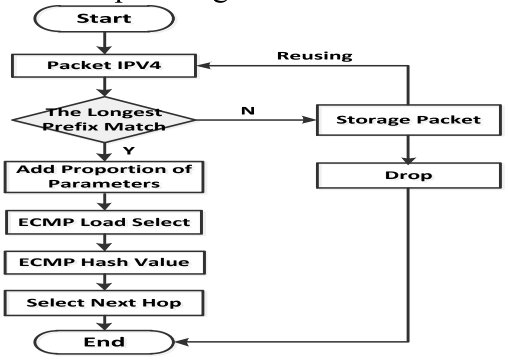

Fig3. ECMP flow

Flow Process

Use the flowlet table shown in Table 2 to record information about the F_id and the N_hop.

Table 2. Flowlet table

F_id	*N_hop*
23	1
45	2
…	…

In order to avoid the reordering of packets in TCP, a flow threshold T_f is set in the study, indicating that the time interval between two consecutive packets, usually hundreds of microseconds of orders of magnitude. If it is detected that the idle interval between the two burst traffic of the packet is greater than the traffic threshold value T_f, when the data packet is continuously transmitted, the packet is transmitted along the path different from the first non-output reordering packet flow, which ensures that traffic is delivered on different paths while still arriving at the receiver in sequence, so as not to lead to the reordering of packets.

When the first packet of the flow arrives at a switch, firstly, the five-tuple hash value of the flow is calculated by the improved ECMP and the current F_id is recorded, creating an entry in the hash indexed flow table. And then by the current F_id and hash values for selecting the best next-hop forwarding the flow. The switches to find the purpose of packet IP address, the best next hop count are stored in the flowlet table N_hop. And use it for all subsequent packets of the forward process.

When the second packet arrives, the switch looks up the flow table entry, and calculates the processing time. The switch obtains the timestamp of the current packet, and determines whether the time interval between the current flow and the last time flow exceeds the threshold T_f. If it is exceeded, the F_id value of the packet is updated to indicate that the packet belongs to a new flow. Subsequently, the switch select the current minimum hop count from the flowlet table as a new flow for forwarding. Otherwise, it is recorded in the flowlet table and as the best next hop for the flow. The switch replaces the old flowlet entry with the current best hop, and then forward the flow. This is followed by the forwarding of the flow in turn. In the flow forwarding process, different flows are distinguished by F_id. Among them, the data flow field state transfer diagram shown in Figure 4.

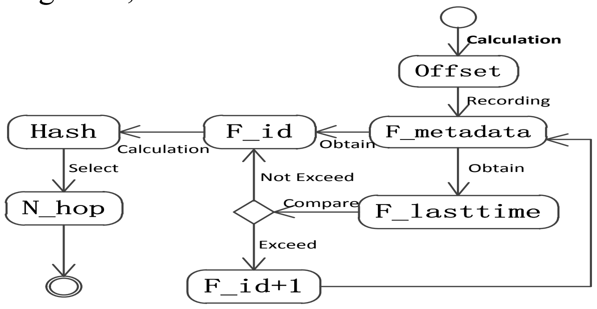

Fig4. Data flow field state transition figure

P4 Language Programming

The offset of the current packet is calculated by the five-tuple hash value in the flow table and is recorded in the metadata. At the same time with two registers to store the current data packet timestamp F_lasttime and flow F_id information. The access timestamp of the current packet is then accessed by accessing the resulting F_id instance and the F_lasttime instance in the offset access register. Finally, the switch forwards the path according to F_id and F_lasttime in the packets.

Using P4 language data plane switching flow processes and the path forward. The data header format is similar to the IP header. The metadata contains the definition of the special status field, the definition fields are as follows:

The metadata definition

```
header_type F_metadata {
    fields {
        F_times : 48;  // Record time threshold
        F_ipg : 48;  // Packet gap
        F_lasttime: 48;  // Timestamp
        F_id: 16; // Flow ID
        E_offset: 14;  //ECMP's offset
        Nhop_ipv4: 32;
        }
    }
```

P4 processing is the key flow control program, the main codes are as follows:

The flow control program

```
control ingress {    // Flow control procedure
    apply(flowlet);    // Get the current flow state information
    if (F_metadata. F_ipg>FLOWLET_TIMEOUT)
    {   // Determine whether the timestamp threshold is exceeded
    apply(new_flowlet);  // F_id+1
        }
    apply(ecmp_group);    // ECMP equivalent routing
    apply(ecmp_nhop);  // The next hop
    apply(forward);   // Flow forward
    }
```

Among them, the path selection and the next-hop route action tables are defined as follows:

Action table

```
table ecmp_group {
    reads {
      ipv4.dst_Addr : lpm;   // The longest prefix match
       }
    actions {
      drop;
      ecmp_select;   // ECMP routing
       }
    }
table ecmp_nhop {
    reads {
      F_metadata. E_offset:exact; // ECMP offset
       }
   actions {
      drop;
      f_nhop;  // Select the next hop router
       }
    }
```

Simulation Results and Analysis

NS2 is a discrete event-driven object-oriented network simulator that includes a simulation event scheduler, a network component object library, and a network builds model library [18]. NS2 as one of the network simulation tools. It's great significance in terms of network protocol simulation research [19]. On the one hand, through the preparation of scripting program on the implementation of the agreement and algorithm simulation for the network planning and evaluation of network performance to provide a reference. The other hand, through the preparation of the new C ++ and Otcl codes compile them into the NS2 kernel to achieve network protocols and algorithm development.

In this paper, the feasibility of improved ECMP is simulated by NS2 simulator, under the same conditions, compared the performance of end-to-end delay, packet loss rate and throughput with the traditional ECMP. Simulation of three layers of fat tree topology shown in Figure 5, two core switches connect two areas, each access switch to connect two servers. Each link has a bandwidth of 10M and a full-duplex mode. Each link queue mode is set to drop-tail, the queue size is 1000, the link delay is 0.5 ms, and the traffic is TCP , UDP random mixed flow, simulation time is set to 5s, which data acquisition and analysis of interception 0 ~ 4.5s of data, to facilitate the analysis of data results.

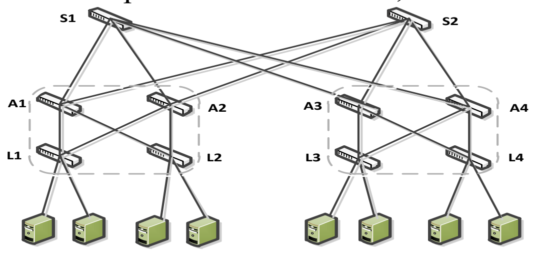

Fig5. Simulation of topology

Performance comparison is shown in Figure 6: (a) shows the end-to-end delay with time changes figure, at the beginning of the simulation, the two algorithms are similar to the delay, which is due to the beginning of only a small amount of data packets for transmission, time is relatively stable, but after 1.0s, with the increase in data packets, making a variety of data packets began to seize the network bandwidth, delay time began to fluctuate. The traditional ECMP delay fluctuates greatly, especially at 1.5s, close to 0.18s, indicating that there is a hot path at this time, the uneven traffic resulted in a large delay, while the improved ECMP delay is always below 0.07s relatively stable. It can be concluded that the improved ECMP is better than the traditional ECMP in terms of end-to-end delay. (b) shows the packet loss rate at different times. With the increase of time, the traditional ECMP packet loss rate gradually increased, indicating that the link has occurred congestion, packet loss is more serious. While the improved ECMP packet loss rate is more stable, and smaller than the ECMP packet loss rate. This shows that improved ECMP performance is better in the packet loss, can effectively reduce network congestion. (c)shows the change in network throughput over time. As a whole, it can be seen that under the same conditions, the throughput of the two images are similar, and the average is maintained at 1Mb/s, but the improved ECMP throughput is slightly higher than the traditional ECMP throughput, and relatively stable.

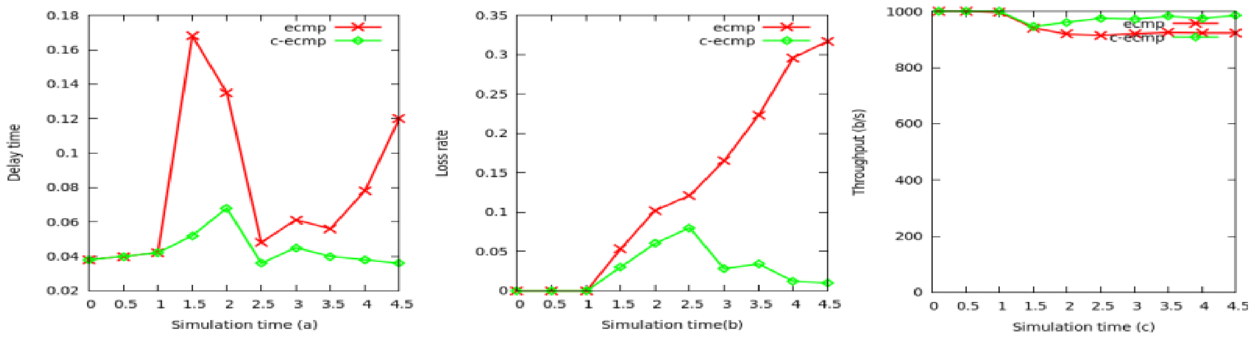

Fig6. Performance comparison

Through the comparison of end-to-end delay, packet loss rate and throughput, there are some fluctuations, but the improved ECMP fluctuation is small. As can be seen from the figure, under the same conditions, improved ECMP can lower the end-to-end delay, reduce the packet loss, improve network throughput, can effectively adjust the network state, to achieve load balancing.

Conclusion

Drawing on SDN network data plane characteristics and P4 language programming, this paper proposes programmable load balancing based on multipath. The forwarding process of the data plane flow is achieved through the P4 language, without flow table rule requesting control plane, and the improved ECMP is used in the path selection to avoid the limitation of being allocated to the same path under multiple operation flow. Using P4 language design data plane flow processing process can save cost, shorten the development cycle. The design adopted the flexible path selection strategy to avoid the link congestion, and achieved the load balancing of the network. Finally, the feasibility of the improved ECMP has shown through simulation to reduce delay and the packet loss rate, improve the network throughput, and perform better than the traditional ECMP. But there are also some limitations such as the flow allocation of the proportion of the selection parameters, build the simulation topology, performance evaluation parameters, simulation time settings and other aspects of consideration is not perfect, all of those can be further improved.

With the continuous development of the data center network, the research on data plane programmability and load balancing will be further improved, especially in terms of flexibility and scalability. Because there is a certain similarity between the traditional network and the SDN network, we can combine the characteristics of the two networks. With the help of machine learning and big data techniques, classic algorithms in traditional networks can be adapted for SDN, load balancing problem can be effectively handled thereby improve the overall performance of the network.

References

[1] Lei Baohua, Wang Feng, Wang Qian. SDN core technical analysis and practical guidance [M]. Beijing : Publishing House of the electronics industry, 2013.

[2] SDNAP. Comprehensive comparison of four core architecture [EB/OL]. http://www.educity.cn

/it/Cisco/201002261625451686.htm,2014-3-22.

[3] Song H.Protocol-oblivious forwarding: unleash the power of SDN through a future-proof forwarding plane[C]. //Proc of the 2nd ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking (HotSDN'13). New York:ACM,2013:127-132.

[4] Jeyakumar V,Alizadeh M,Kim C.Tiny packet programs for low-latency network control and monitoring[C]. Proc of the 12nd ACM Workshop on Hot Topics in Networks,2013:1-7.

[5] Zhu S, Bi J, Sun C,et al. SDPA: Enhancing Stateful Forwarding for Software-Defined Networking[C].//Network Protocols 2015 IEEE 23th International Conference on, 2015:323-333.

[6] Intel FlexPipe. http://www.intel.com/content/dam/www/public/us/en/documents/product-briefs

/ethernet-switch-fm6000-series-brief.pdf

[7] XPliantTM Ethernet Switch Product Family. http://www.cavium.com/XPliant-Ethernet-Switch -Product-Family.html.

[8] Cisco highlights next big switch. http://www.biztechafrica.com/article/cisco-announces-next -big-switch/5448/#.VP4mCYWltVZ.

[9] Sivaraman A,Subramanian S, Alizadeh, et al. Programmable Packet Scheduling at Line Rate[C]. Proc of the 2016 ACM SIGCOMM Conference, 2016:44-57.

[10] Sivaraman A, Cheung A,Budiu M, et al. Packet Transactions: High-Level Programming for Line-Rate Switches[C]. //Proc of the 2016 ACM SIGCOMM Conference,2016:15-28.

[11] Jain S, Kumar A, Mandal S, et al. B4: Experience with a globally-deployed software defined WAN[C]. //ACM SIGCOMM Computer Communication Review. ACM, 2013, 43(4): 3-14.

[12] Kaur S, Singh J. Round-robin based load balancing in Software Defined Networking[C]. //Proc. 2nd International Conference on Computing for Sustainable Global Development, 2015:2136-2139.

[13] CX Cui, YB Xu. Research on Load Balance Method in SDN[J]. International Journal of Grid and Distributed Computing,2016,9(1):25-36.

[14] Cohen R, Lewin E L, Naor J S. On the effect of forwarding table size on SDN network utilization[C]. // Proc of IEEE Conference on Computer Communications, 2014:1734-1742.

[15] Xu X, Cao L, Wang X. Adaptive Task Scheduling Strategy Based on Dynamic Workload Adjustment for Heterogeneous Hadoop Clusters [J]. IEEE Systems Journal, 2016, 10(2):471-482.

[16] Bosshart P, Daly D, Gibb G, et al. P4: Programming protocol-independent packet processors[C]. //ACM SIGCOMM Computer Communication Review. ACM, 2014, 44(3):87－95.

[17] Alizadeh M, Edsall T. Conga: Distributed congestion-aware load balancing for datacenters[C].//ACM SIGCOMM Computer Communication Review, 2014, 44(4):503－514.

[18] Wang Hui. NS2 Network Simulator of theory and application[M]. Xi'an:Northwestern Polytechnical University Press , 2008.

[19] Ekram H,Issariyakul T. Introduction to Network Simulator NS2[M]. Springer, 2009.