Research Journal of Applied Sciences, Engineering and Technology 5(7): 2451-2457, 2013
ISSN: 2040-7459; e-ISSN: 2040-7467
© Maxwell Scientific Organization, 2013
Submitted: July 26, 2012 Accepted: September 03, 2012 Published: March 11, 2013

Research on Performance between IP and Vector Forwarding

Aqun Zhao, Hao Zhou and Yugang Hao
School of Computer and Information Technology, Beijing Jiaotong University, Beijing, 100044, China

Abstract: With the rapid development of network size and great increase of Internet traffic, it has become a crucial work to improve the forwarding performance of the core devices of Internet or the routers. In this study, the analysis and comparison of instruction cycles occupied by the machine in the process of route lookup in IP forwarding and vector forwarding were carried out which may determine the forwarding performance of a router. The simulation experiments were also made to study the forwarding efficiency of IP forwarding and vector forwarding. The theoretical and experimental results prove that the vector forwarding method is more efficient than IP forwarding which can provide powerful evidence to the application of vector network in the Internet.

Keywords: Forwarding performance, IP forwarding, route lookup, vector forwarding

INTRODUCTION

With the continuous development of science and technology, Internet, telephone integrated services rapidly spread in popularity. And people use computer in every field of our life. But the following is the rapid development of network scale and the sharp increase in data traffic. So the network suffers great pressure.

Router is the core device in the whole internet. So it must have ability to connect different networks, divide the whole network logically using specialized software protocol and filter network traffic.

At the same time, the router can select the information transmission line and choose a shorter path. Thus, it can greatly improve the communication speed and reduce the network load. The result is that it can help the network save resources and improve the network flow rate, so that the network can be more efficient.

Large scale communication data bring a big challenge to the internet. With the growth of communication data, in order to maintain its normal speed of traffic and keep the data not lost, we can find that the performance of the router to forward packets is the key to solve it. So it is important for us to research it.

In the process of routing and forwarding, it is most time-consuming in routing table's lookup, establishment and maintenance. So, the routing table can not only affect the stability and scalability of the internet, but also be an indicator of high-performance router. At the same time, it is natural that routing table becomes an important and difficult point for the design of the router.

From the birth of Internet to now, IP network dominate the entire internet and we can find IP network everywhere of the world. So the study on the router is mainly based on IP network in domestic and foreign. But with some new branches of applications in the network, the information communication requires a higher load. At this time, the conception of vector network is proposed by Beijing Jiao tong University and the research on vector-based exchange equipment also become necessity (Zhao and Liang, 2012; Liang, 2009a).

In the vector network, vector forwarding does not need the process of router table's lookup. So it brings a higher time-efficiency and helps us to avoid many research challenges.

The study describes the current popular router model, the comparison between IP forwarding and vector forwarding, the simulation of it and the conclusion.

VECTOR FORWARDING

On the basis of two invention patents "a kind of vector network addressing method" (Liang, 2009a) and "a kind of vector connection establishing method in vector communication network" (Liang, 2009b), a new type of communication network called vector network has been proposed which has many eminent characteristics such as simpleness, fractal, creditability, infinite extensibility, QoS support, self-organization, distributing and full support to mobility and multicasting.

The foundation of the vector network is a sort of new switching address to switch data called vector address. Vector address has made use of a category of new addressing scheme which is coded basing on the ports of communication devices. In detail, a vector address is made up of a finite sequence of sub-addresses. The first sub-address of the sequence is the output port number of the source device. The second sub-address is

Corresponding Author: Aqun Zhao, School of Computer and Information Technology, Beijing Jiaotong University, Beijing 100044, China

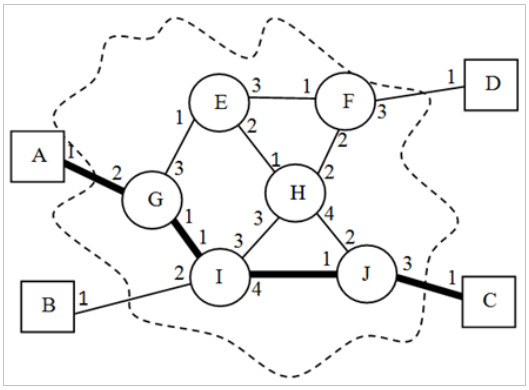

Fig. 1: Vector network and vector address

Table 1: Codes of port and bits occupied node

Node	A	G	I	J
Port number	1	1	4	3
Quantity of ports	1	3	4	4
Number of bits	1	2	3	3
Binary coding	1	01	100	011

the output port number of the first forwarding device on the data transmission path. The third sub-address is the output port number of the second forwarding device on the data transmission path. The rest may be deduced by analogy. In the end, the last sub-address is the output port number of the last forwarding device on the data transmission path. Using above vector address coding method, an arbitrary data transmission path can be coded into a unique vector address. On the opposite, if a source device knows the vector address, it can locate a destination device uniquely and the passed route can be determined completely.

Take the network shown in Fig. 1 as an example to explain the definition of VA. In Fig. 1, the squares and circles labeled from A to K are nodes and the solid lines between pairs of nodes are links. Each node has several ports numbered from 1 which is called port number. A path in the network can be described by a node sequence. For instance, {A, G, I, J, C} is a path from node A to node C. A path can also be described by a sequence of output port numbers. For example, path {A, G, I, J, C} can be denoted as "1143" where digits 1, 1, 4 and 3 correspond to the output port numbers of node A, G, I and J, respectively on the path. Note that the destination node C is excluded. The digital string "1143" is exactly a VA from node A to node C. It is worth noting that VA is directional. The VA from A to C and that from C to A are different.

Formally we define a VA from a source node to a destination node as a sequence of output port numbers of nodes along a path from the source to the destination. The port numbers in the sequence look like direction indicators guiding the packets to arrive at the destination step by step. This is why this kind of forwarding address is named VA. Each port number in VA is called element address.

In practice, VA is expressed in binary format. Also taking Vac = 1143 as an illustration, the results are listed

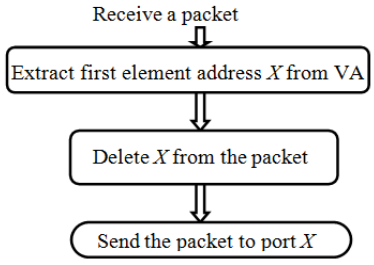

Fig. 2: Forwarding program executed in VS

in Table 1. In the first row of this table the output port numbers of node A, G, I and J are listed respectively. The quantity of ports of each node is given in the second row. According to the quantity of ports, the number of bits to code these ports in binary format in each node can be deduced which is listed in the third row. In the last row the element address of each node in binary format is given. So the final VA in binary format is "101100011".

The following data forwarding method is applied in VN. When a VS receives a data packet from one of its input ports, it extracts the first element address of the VA from the packet, deletes it from the packet and sends the packet to the output port pointed by this element address. The forwarding program executed in VS is described in Fig. 2 where the first element address of the VA is denoted as *X*. This forwarding program executes once when a VS receives a packet.

According to the above definition of VA and the forwarding method in VN, it can be reduced that VA has the following characteristics:

Infiniteness: The number of VA between a source and a destination is not just one, but infinite. In practice, we usually adopt the shortest path or a path close to the shortest. But some cycles may be useful for encryption.

Non-readability: From the forwarding process in VN, it can be seen that a VS just can understand the element address belong to itself. It can't get the frontal element addresses and can't understand the posterior element addresses.

Encryption property: A VA can be encrypted because any VS in the data transmission path can complete its forwarding task as long as it knows how to decrypt its own element address. So long as the source node establishes a secret key with each V in the path through negotiation before communication begins, the communication with address completely encrypted can be realized. With a view to the non-readability, VA has a quite strong ability to realize encryption communication. Indeed, not only data transmitted can be encrypted, but the identities of the communication participators in both sides can also be protected.

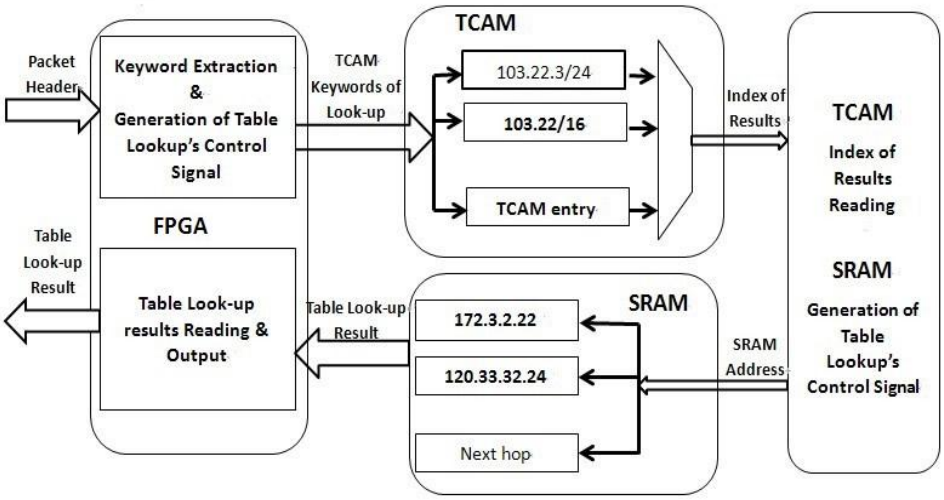

Fig. 3: Implementations of router table lookup with pipeline

Relativity: The meaning of a VA is relative to the source. If a VA is given without the information about the source, its meaning is ambiguous.

Variability in address length: The length of VA is variable. It may be any long as needed.

Possessing routing information: A VA defines a path in the network and contains routing information useful for VS. A VS can get the routing information from VA directly which have to be gotten through table-lookup in IP and ATM networks.

ROUTER MODEL

We choose the mainstream router architecture, i.e., the crossbar-based switch router and then study the routing performance in IP forwarding and vector forwarding (Xie, 2008).

In the router with this structural system, the entire routing table is stored in line card. The line card accepts all the packets arrived in the interface, classifies packets and lookups the router table. Ports in crossbar can erupt simultaneously. System control and router table update is responsible for the CPU (Central Processing Unit).

Router forwarding engine: Parallel lines divided a task into serial sub-tasks, each sub-task runs on a single core and each sub-task is independent (Kurose and Ross, 2009). Many applications require a process order and this cannot use task parallelism or data parallelism. But many of these applications can be paralleled in pipeline parallel.

Forwarding engine needs to do the header analysis, QoS (Quality of Service) implementation, security testing, direct examination, lookup table in unicast and multicast and so on. Parallel processing is in accordance with the various functional modules in the processing order of their relevance and then all the function models are processed parallel.

After the packets enter forwarding engine across network interface, the following steps will happen.

First, convert the input data format into the internal needs of the bus engine format.

Secondly, extract packet header from the data, do the IP packet header inspection, grouping and classification. Push the packet data into the package cache.

Thirdly, classify sub-tasks and parallel input them into routing table look-up, IPV4/IPV6 header processing and special handling after the header is extracted. In the sub-task of routing table lookup, the router mainly do the lookup table of unicast and multicast, priority or safety inspection and so on. In the sub-task of IPV4/IPV6 header processing, it modifies the TTL (Time to Live) parameter, recalculates the IP heard checksum. In the sub-task of special treatment, the IP address management, LAN (Local Area Network) and WAN (Wide Area Network) protocol and the conversion between them.

Fourthly, send the results of the three sub-tasks into a unified package recombinant task. The sub-task can generate new data header and output signal based on different inputs. In the end, data packet and header packet will combine into a new packet.

Lastly, the new packet will be sent to local processor or exchange network and output from the right port. The whole forwarding is over by now.

Router pipeline implementations: Figure 3 just used TCAM (Ternary Content Addressable Memory) pipelining to achieve a routing table's lookup (Zane *et al*., 2003; Netlogic, 2003). The whole process is divided into three sub-tasks.

The first sub-task is "keyword extraction and the generation of table lookup's control signal". In this section, the keyword table needs will be extracted from packet, then input it into TCAM, at the same time, the control signal of TCAM lookup generates. TCAM table's width can be configured as 72, 144, 288 and 576

bits, respectively. For example, the use of TCAM of Net logic NSE5512, its capacity is 512k*72 bits (i.e., if entry width is 72 bits, the capacity (including the mask) as 512 k×72 bit, in fact bar entry to 256 k). In other words, when the TCAM entries are configured for the 144, the capacity of 128k; configured for 288, the capacity is 64 k bar. Currently, the main data width of TCAM is 72 bits; the following analysis is followed by the premise. The TCAM look-up tables time-consuming can be divided into two parts.

First, 'T1' is the time-consuming in inputting the keyword to lookup the table. According to the design of forwarding format, source and destination addresses are stored in the TCAM as a look-up table's keywords. In IPV4, the IP address length is 4*8 = 32 bits, so the keyword of IPv4 is 64 bits including source and destination IP address. So we can conclude that the length of IPV6 is 256 bits. When the TCAM stores IPV6 prefix table entry, the look-up table clock cycles required T1 = floor (256/72) = 4. Similarly, when the TCAM stores IPV4 prefix, the required clock period T1 = floor (64/72) = 1.

Second, get the result of using keyword to search internal table of TCAM, referred to as the waiting time of look-up table. At present, the industry's TCAM waiting time is usually 10 clock cycles.

Therefore, the completion of the first sub-stage of a mission, for IPV4, will spend T = T1 + T2 = 1 + 10 = 11. But for IPV6, it will spend T = T1 + T2 = 4 + 10 = 14.When the sub-task is over, the index of results of look-up table can be output, so that we can find the result in SRAM.

The second sub-task is "read the result of TCAM index and look-up SRAM table's signal". The phase will first get the index from the first task, then generate the read and write signal according to the index. Under normal circumstances, when the stage is completed, there will still be two clock cycles to be waited, so that we can get the final result.

The third sub-task is "read and output the result of looking up the table". In this phase, we should get the read address and read signal first that generate in the second task, then locate it and read the final result, finally, input the result to the task of restructuring packet as Fig. 3.

FORWARDING PERFORMANCE

There are two solutions to solve different cycles in pipeline's sections. First, we can parallel process the key section to shorten the delay time in the key section; The second is to increase non-critical segment connecting the latch clock latch stages, making the latch time delay and non-critical section corresponding to the segment and equal to the delay t of the key segments of the pipeline cycle. We choose the second way in the study.

The biggest difference between IP forwarding and vector forwarding is that in vector forwarding, there is no need to look up the table. Additionally, the length of vector and IPV4 is equivalent. So vector forwarding

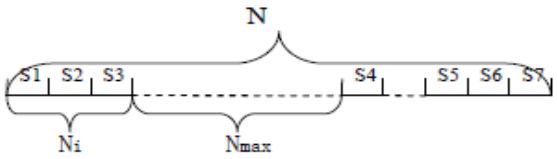

Fig. 4: Definition of sub-tasks

procedure will experience only 1, 2, 6, 7 sub-tasks, but IP forwarding will experience the whole seven sub-tasks. Because the length of vector and IPV4 is equivalent, the time-consuming in 1, 2, 6, 7 sub-tasks are the same. So vector forwarding can save time in 3, 4, 5 sub-tasks for the IP forwarding.

For IPV4, a packet forwarding needs N = 10 + 2 + 7*2 = 26; but for vector forwarding, because the key section of the pipeline is S6, there is no routing table look-up, so 3, 4, 5 sub-tasks are not needed and the waiting cycle after 3 and 4 sub-tasks are also not needed. Thus, a packet forward needs M = 4*2 = 8 in vector forwarding. It can be seen that completing packet forwarding in vector network can save 26 - 8 = 18 clock cycles relative to IP network. At this point, we can see, using the vector forward mode of clock cycles consumed only about the use of IP (IPv4) forwarding the number of clock cycles consumed 1/3.

Assuming a clock cycle is T (ns); router line card speed is V (Gbps); the packet length is L (bits); In the forwarding process, serial operation of all sub-tasks and waiting section together time-consuming is N; The most time-consuming in all the process is Nmax; all the sub-tasks before Nmax is Ni; All of the key sub-segments of the task time-consuming is Nm.

Some variables above are shown in Fig. 4, there are total seven sub-tasks named S1-S7 and the dashed part represents waiting time before the last sub-task is over.

The time for a router to receive a complete packet can be denoted as:

$$t = 8L/V \tag{1}$$

The number of data packets to be transmitted successfully per second can be denoted as:

$$N_{suc} = \begin{cases} (10^9 - NT)/N_m T & N_{\max} \le N_m \\ (10^9 - (N_i + N_{\max})T)/N_{\max} T - \\ \text{ceiling}((N - N_{\max} - N_i)/N_{\max}) - 1 & N_{\max} > N_m \end{cases} \tag{2}$$

We set a clock to be 10 ns and router line card speed is 10 Gbps. And we assume that the shortest multicast packet length is 40 bytes.

According to the above formula, we derived that: the router receives a complete packet and the time consumed is 40*8 bit/10 Gbps = 32 ns, so in one second, the number of packets reached to the router is n = 1s/32 ns = 31250000. When we process a header of IPV4, we can know the number of data packets forwarding successfully in a second is (1s - 16*10 ns) / (10*10 ns) =

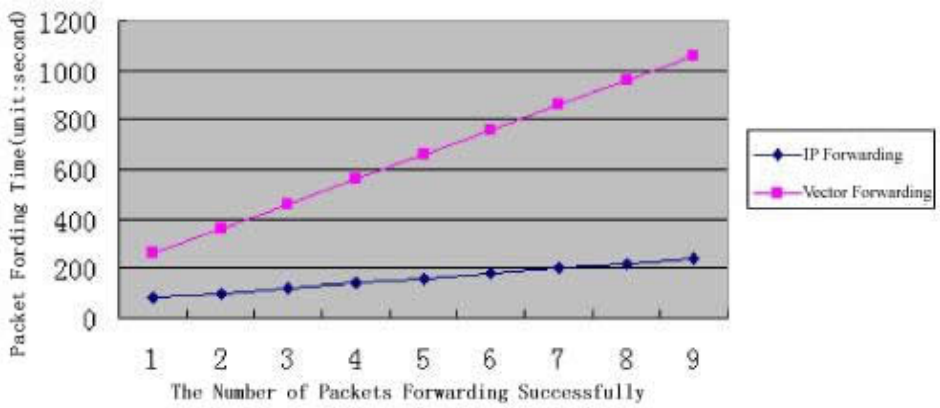

Fig. 5: Forwarding performance comparison

9999998, now the number of packets in cache can reached 31250000-9999998 = 21250002; thus, forwarding efficiency = (9999998/31250000) *100% = 0.3199; while in the vector forwarding, the number of data packets forwarding successfully in a second is (1s - 8*10 ns) /2*10 ns = 49999996. As the time that router forward packets less than the time that router received packets, all the packets received by router can forward successfully per second, which means forwarding efficiency is 100%.

We can conclude that packets forward in vector network can forward more successfully than in IP network. It can process 49999996 - 9999998 = 39999998 more packets/sec. So vector forwarding is more efficient than IP forwarding.

When the packet length is 1500 bits, the time router receives a complete packet can be denoted as:

$$t_1 = 1500 * 8(bit) / 10(Gbps) = 1200(ns) \quad (3)$$

If the packet is IPV4 packet, it will spend t2 = 260 ns completing forwarding a packet. So when the router accepts the next packet, the previous has sent out. Thus, forwarding efficiency is 100%.

For a single packet, IPV4 forward time is 26*10 ns = 260 ns; vector forward time is 8*10 ns = 80 ns. So we can find that to forward a single packet, vector forward will save t = 260 - 80 = 180 ns relative to IP forwarding.

Set the successfully transmitted packet number n as the X-axis, the time it needs as Y-axis, then we can get Fig. 5.

Analyzing Fig. 5, we can conclude that for both forwarding, the forwarding efficiency is related to many factors. For example, packet length L; set T as time which router accepts packet minus time which router forwards a single packet, T = (t1 - t2). When the rate of packet receiving is fixed, if T>0, forwarding efficiency is same as in IP forwarding and vector forwarding; but if T<0, forwarding efficiency is much higher than IP forwarding.

From another perspective, the vector can be forwarded to the router's maximum line cards speed even if packet length takes the shortest, forwarding efficiency can also reach 100%; but to IP forwarding, generally forwarding efficiency can't reach 100%, that is to say, there are still packets accepted by router but not forward successfully. These packets must be stored in the cache. In this case, because IP forward efficiency can't reach 100%, with more and more packets reaching router and need to be forward, we must set cache in order to prevent packets being lost. Thus, a new problem will be occurred. If the cache set is too small, the packets must be lost under certain circumstances. Or if we make the sender resend the packets, this will result in the packet network congestion. If the cache is too large, in most case, the cache utilization is very low and it results in resources' waste. While, if you use vector forward, these issues will not appear. So we can know that vector forwarding can save not only time but also resources.

SIMULATIONS

Simulation is conducted using OPNET simulation software, the experiment is a simulation of the forwarding nodes, respectively, in the process model and control node model (Chen, 2004).

Figure 6 and 7 are the results of IP forwarding and vector forwarding, which the horizontal axis represents the length of the packet, the vertical axis represents the forwarding efficiency.

We can see from the two test results, for IP forwarding, with the packet length increasing, forwarding efficiency becomes more and more increasing. When the length reaches a certain value, forwarding efficiency becomes 100%. But for vector forwarding, it keeps 100% in all the above cases.

From the experimental results, we can conclude that the vector forwarding is more time-effective than IP forwarding.

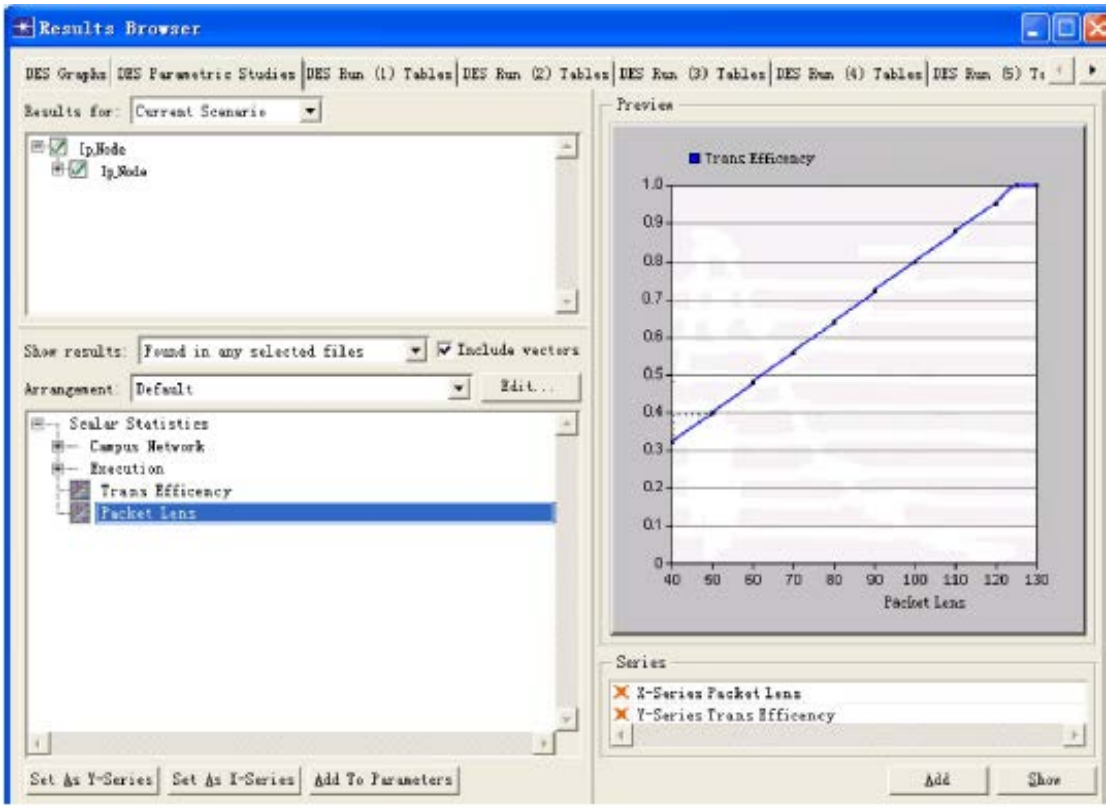

Fig. 6: The efficiency of IP forwarding

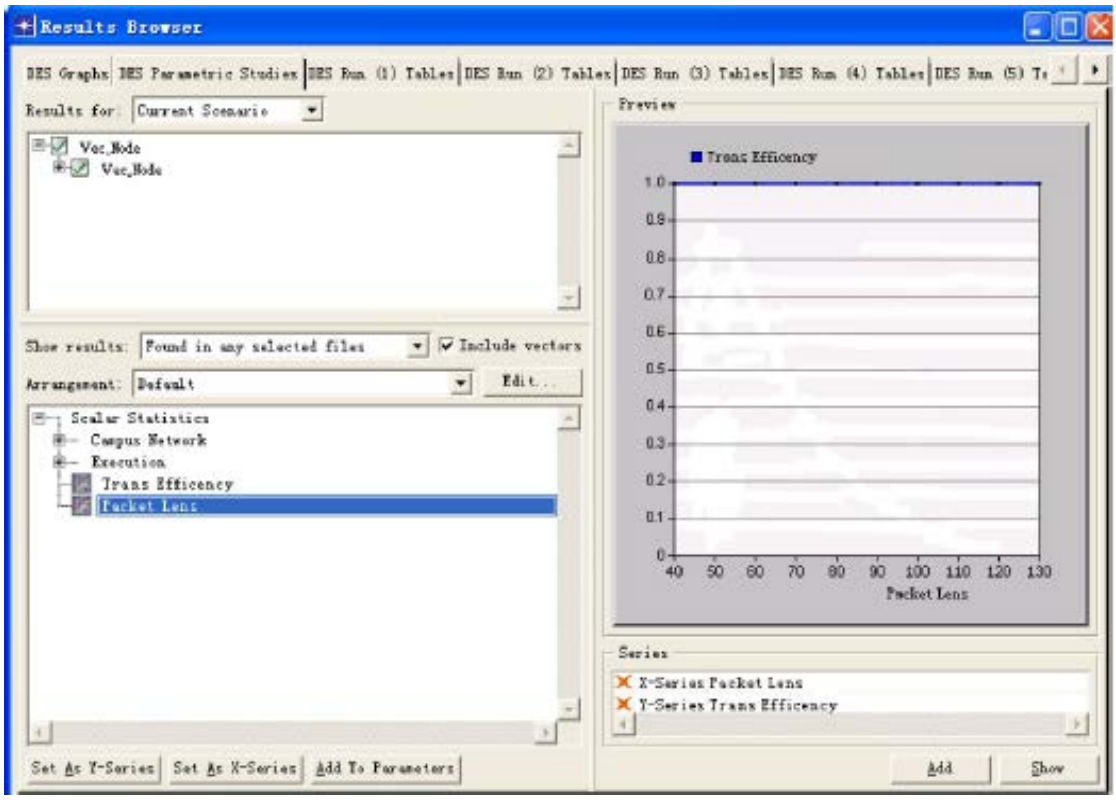

Fig. 7: The efficiency of vector forwarding

By the two figures, we can also see that when the packet length is lower than 110 bits, IP forwarding efficiency is lower than 90%, or even reach 50%. In this stance, packets that unsuccessfully forward must be processed, store in the cache or discarded then retransmitted by sender.

If we set the cache, for the forwarding efficiency of less than 50%, there will be more than half of the packets will be stored in the router. And at this time, packets will be blocked thousands or more in one second in the router. In order to prevent these packets from losing, the cache capacity will be a large consume.

If we take the way of discarding packets, the vast majority of data packets in the network will be retransmitted. The entire forwarding process will become very slow, or even result in paralysis of the entire network.

For the vector forward, since the forwarding rate is always 100%, that is to say, it can handle all the packets in time, so we don't have to consider both cases above.

From theoretical analysis and experimental results, we can both draw the following. When the speed router receives a packet's is fixed, the shorter packet length is, the higher forwarding efficiency in vector forwarding to IP forwarding. When the graphics rate router receives a packet increases one or more grades, IP forwarding will be too late to deal with the current received packets, but vector forwarding is possible. Increasing line card speed is the trend and with the rapid increase of line card speed, the advantage of vector forwarding becomes increasingly significant. Therefore, the research of vector forwarding becomes the trend.

CONCLUSION

We can draw the conclusion that vector forwarding is more time-effective than IP forwarding from the research on the transmit efficiency of vector forwarding and IP forwarding in the routing process. The reason is that the vector forwarding eliminates the process that consumes most and route table's look-up. Coupled with the inherent nature of vector address, vector network has more obvious advantages than IP and ATM network. It not only improves the timeliness, but also enhances the security and reliability in the process of transmitting information.

ACKNOWLEDGMENT

This work was supported by the Fundamental Research Funds for the Central Universities (No. 2012JBM025) and in part by the Open Research Fund from Key Laboratory of Computer Network and Information Integration in Southeast University, Ministry of Education, China (No. K93-9-2010-08).

REFERENCES

Chen, M., 2004. OPNET Network Simulations. Tsinghua University Press, Beijing, China.

Kurose, F.J. and K.W. Ross, 2009. Computer Networking: A Top-Down Approach Featuring the Internet. Higher Education Press, Beijing, China.

Liang, M.G., 2009a. A Method for Vector Network Address Coding. China Patent, ZL200610089302.6, (In Chinese).

Liang, M.G., 2009b. A Kind of Vector Connection Establishing Method in Vector Communication Network. China Patent, ZL200710064804.8.

Netlogic, M., 2003. Ternary Synchronous Content Addressable Memory (IPCAM). Retrieved from: http:// www. netlogicmicro. com/pdf/NL82721.pdf.

Xie, X.R., 2008. Computer Network. Publishing House of Electronics Industry, Beijing, China.

Zane, F., N. Girija and A. Basu, 2003. Coolcams: Power-Efficient TCAMs for forwarding engines. IEEE Infocom, 1: 42-52.

Zhao, A.Q. and M.G. Liang, 2012. A new forwarding address for next generation networks. J. Zhejiang Univ. Sci. C, 13(1): 1-10.