RESEARCH ARTICLE OPEN ACCESS

The Research on Accelerated Routing Lookup Algorithms

Yaqiong Li[1], Xiaohua Meng[1], Jie Li[2]

[1](Department of Computer Science,Jinan University, Guangzhou 510632,China)

[2] (College of Software, Henan University, Kaifeng, Henan 475001 ,China)

----------**********----------

Abstract:

With the rapid development of the network, the speed of optical fiber and interface transmission has been improved, and the ability of network equipment to handle message processing has become the main bottleneck of the current high performance network development. Core routing turned sharply increasing scale of published, leading to increasing demand time and memory look-up table, and the need to press the longest prefix match when forwarding lookup, make address lookup in numerical and length on the two dimensions. Therefore, to deal with a large number of packets and ensure network quality, faster routing search speed is needed. Surrounding the technology of high performance routing lookup is studied, this paper puts forward two kinds of routing lookup algorithm based on GPU acceleration technology, based on hash table(CUCKOO FILTER) lookup algorithm of acceleration and LCTrie acceleration of tree search algorithm, and compared and analyzed the advantages and disadvantages of the two methods.

Keywords **—Routing Lookup, GPU, CUCKOO FILTER, Hash Table, LCTrie, Fast Routing**

----------**********----------

I. INTRODUCTION

At present, the development of network is exponential growth, and the router has higher requirements for the adaptability and forwarding ability of large-scale transfer. Therefore, it is necessary to propose new solutions for IP routing.

Traditional routing has hardware based routing and software-based routing. The hardware based router is a matching of hardware implementation prefixes, such as the 24-8 DIR algorithm[1], based on cache[2] and based on TCAM [3].They usually applies to provide high capacity for a particular application, however, due to a hardware algorithm performance depends on the performance of the hardware itself completely, but high performance hardware price is more expensive, it is difficult in the current network popularization, and it can't meet the demand of the application of high speed programmable. On the other hand, software based routing usually has: 1. Based on the IP lookup of binary Trie tree, relevant articles are RadixTrie [4], Patricia, Multiway and Multicolumn [5], etc. The main idea is based on IP address prefix binary bit tree structure, when routing lookup traverse the binary tree, until you reach the longest branch, can no longer continue downward, the current position as the destination IP of the longest prefix match. 2. Based on the algorithm of multi-branch tree search, the typical algorithm includes the control prefix extension algorithm [6], dictionary lookup algorithm, LCTrie lookup algorithm, etc. Binary Trie tree binary search can reduce the general search space each time, while the multi-branch tree lookup can reduce the retrieval time and space more quickly. 3. Based on hash table lookup, due to its

simplicity, it becomes the classic route finding algorithm of Linux operating system. However, most of the traditional dynamically programmable software routing can only support the throughput of less than 10Gbps, which is a challenge to meet the current high throughput network requirements. The reason is that the existing algorithms are more serial, and rely on CPU computing power, which can only run on one kernel in a multicore processor even if multi-core parallelism is used. To solve this problem, we can only select algorithm optimization and parallelism. GPU, as a graphical processing device, has very high parallelism. In this paper, GPU is used as the research of general calculation, and the routing algorithm is transplanted to GPU to do parallel computation to realize acceleration.

In this paper, by using Graphics Processing Unit (GPU) as a support infrastructure of routing retrieval algorithm, using multi-threaded characteristics of heterogeneous computing, batch Processing of routing information, instead of the usual bunch of structure, process one by one of the packets. By experiment, the algorithm based on GPU parallel acceleration is used to compare the route retrieval based on CPU processing, which accelerates nearly 30 times, which proves that GPU architecture is adopted as the executable of routing retrieval. In the following experiment, two different kinds of retrieval algorithms are used to compare the efficiency of route retrieval. The experimental results show that the algorithm based on LCtrie route retrieval is better than hash retrieval on GPU. This paper for the GPU architecture the feasibility of the application on the routing retrieval is analyzed, and then choose from a variety of software routing retrieval two as experimental research method, the third part introduces the whole process of the system, emphatically introduces the concrete realization method of algorithm on GPU. Finally, this paper makes a performance evaluation of the solution and analyzes the results of the two algorithms.

II. THE BACKGROUND INFORMATION

A. The architecture and applicability of GPU

GPU is designed for large throughput. GPU features a small number of cache and a large number of ALU logical units. Unlike CPU, GPU uses caching not to save the data that needs to be accessed later, but to improve the service for threads. If there is a large number of threads that need to access the same data, the GPU cache merges these same access, then access the dynamic random access memory DRAM, and after obtaining the data, the cache posts the data to the corresponding thread. But due to the need to access the DRAM, a certain delay was created. GPU control units can combine multiple accesses into less access. Although GPU has DRAM delays, there are a lot of ALU and a very large number of threads, which can balance the problem of memory latency and make full use of the multiple ALU features to achieve a very large throughput effect.

The CPU is suitable for executing sequential code, GPU is suitable for data parallel code, and GPU for general-purpose computing has the following advantages: 1. Computing power is several times or even tens of times that of the mainstream CPU. 2. Hardware multithreading, high memory bandwidth. 3. The requirement for memory bandwidth is higher than delay. 4. Masking memory access latency through concurrent access to multithreading. 5. Have a large number of parallel computing resources to handle different data. 6. Almost zero-consuming thread creation and switching

In the process of routing retrieval, a large number of packets are routed to the query, and the process itself has high concurrency. So in this paper, using the GPU parallel computing, the processing of speed routing lookup, mainly studies the hash table

lookup algorithm and multiple branching tree search algorithm, selection of an optimal algorithm for the GPU parallel speed up processing. Mainly using CUDA model, CUDA defines the framework and basic programming model of Nvidia series GPU. In this paper, Nvdia GeForce GTX750Ti is used as the research platform.

B. The Routing lookup based on hash table

Based on hash table lookup, the algorithm's main idea is to map the data into its first hash value, through calculating the storage address of data elements to find a way, is a kind of O (1) search, the efficiency is very high. In some huge amounts of data processing and cloud computing technology often used a hash table lookup algorithm, for processing large amounts of data, often need a index data structure, to quickly determine whether records exist, this method can also be used for routing lookup. Common algorithms are: BLOOM FILTER [8] and CUCKOO FILTER. BLOOM FILTER is a bitmap hash method, with high spatial utilization. When the value is inserted, the hash function is mapped to different bit positions and set 1. Only the bit bits that are mapped to by all hash functions are found in the lookup to indicate that this value exists in the hash table. There are two problems with this model: 1. False positives can only be provided in the query when they can exist and must not exist. 2. Missing the report, if the element is deleted, the bit being mapped to is set to 0, which can not find the other value that been matched in the same bit when retrieved. To solve this problem, this article used the CUCKOO FILTER [10].

C. CUCKOO FILTER

CUCKOO FILTER was introduced in 2001 by Rasmus Pagh and Flemming Friche Rodler. The hashing method is to solve the conflict of hashing, and to use less calculation in exchange for larger space. The name derives from the fact that the hash method ACTS like the cuckoo's laying eggs in other nests and squeezing other birds' eggs. It has the characteristics of small space and quick query. For BLOOM FILTER, CUCKOO hashes support quick queries and delete. Its hash function to is in pairs, and there are 2 barrels (bucket), each of the elements, through the two hash function, and mapped to the two barrels of different location, a record, a standby position, standby position to use when dealing with collision.

The CUCKOO's method of handling collisions: kick the elements that used to be occupied, and the kicked elements can be placed in a spare position. If there is an element in the position, and kick the element out, it will find another hash of it, so that it can be repeated until the number of kicks is up to the limit and the task is full. 1. For key hash, you get two hashes of k1 and k2, and if one of them is empty, insert it directly. 2. Otherwise, insert at any one position and kick out key of the original position. 3. The kicked key needs to be re-inserted until no key is kicked out.

D. The Routing lookup based on LCTrie

The Route retrieval is based on LCTrie lookup. The LCTrie hierarchical compression tree, which is a path compression based on the binary Trie tree. Binary tree is a bit tree, left the child to 0, the right child is 1, usually binary tree will have a lot of empty nodes no routing entry inside, causing the height of the tree is very high, in order to improve the efficiency of the query, you can use path compression, remove the empty nodes, combination of a single node of the path, and then optimize, became LCtrie, as shown in figure 2.1 for LCtrie, is a more variable branch branch trie tree, search process is similar to a binary tree, just by multiple bits are traversed at once. LCtrie reduced the height of the tree, and in all algorithm based on tree, tree height and determine the need to access the memory, the number of times to fetch the time-

consuming decided the time performance of the algorithm, so the LCtrie algorithm is a kind of better.

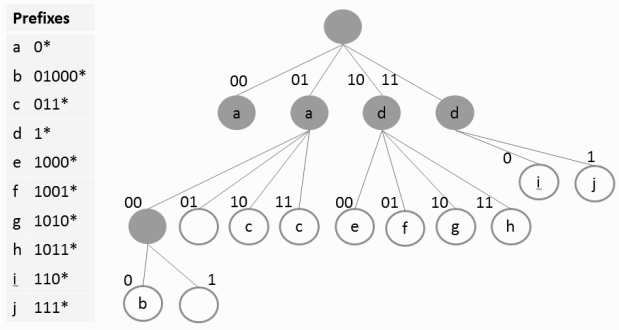

Figure 2.1 LCtrie

About the LCtrie search, the process as well as letters, when the ordinary letters send to the destination, at the first look at the country, if the letters sent from the United States, decided to send to which country, the destination is China, and then match the province, found that to be sent to GuangZhou, letters arrived in GuangZhou, then to regional, streets, and the last to the door, until delivered to the destination. Trie tree, and the structure similar to above, actually for trie tree, to detect the binary bit stream, have no fixed division, the need to detect the location is not fixed, expressed with pos, and the length of the detection index is not fixed, by bits, said each testing point as CheckNode, among them, the CheckNode node contains pos and bits. A leaf node is a routing item, each of which is a leaf. When constructing a tree, the width and depth of the tree should be a reasonable proportion. If the tree is too thin, can be compressed, such as the three nodes compressed into a node, this means that the previous reuse of 1, 2 nodes will be overwritten, when no 1, 2. If it's too fat, it's elongated, the inverse of the process.

III. THE IMPLEMENTATION OF THE ROUTER ACCELERATION

A. Process Design

the routing retrieval system involves multiple steps, including packet receiving, protocol parsing, routing retrieval, and packet forwarding. Using multithreading, for various parts of the different types of thread, use of the CPU's assembly line work: each thread repeatedly perform the same operations on the data set, and the operation result is passed to the next step to other threads. The whole process consists of three parts: 1. Preprocessing: (Receiver thread) for packet resolution, memory allocation and recovery. 2. GPU processing: (Scheduler thread, guaranteed synchronization) scheduling, operation of hash table, query, insert, delete. 3. Subsequent processing: (Sender thread). The assembly line workflow is shown in figure 3.1. The Pre phase executes the Receiver thread; Implementation of Scheduler threads in GPU phase; The Post stage executes the Sender thread

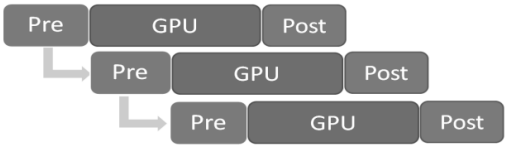

Figure. 3.1 pipeline workflow

In the Pre stage, in addition to receiving and preprocessing the packets, you need to prepare the data for the second phase of the GPU. The CPU needs to first open up the memory space for the GPU and transmit data to the device end. 1. Data preparation: first of all by the table to sort, according to the routing table to segment the prefix length of sorting, network prefix the longer the destination IP address is, the more, that is a 32-bit subnet mask all the IP address in the IP address of the 31 subnet mask before, the destination IP address and the corresponding after sorting the next-hop address pairs of become the new routing table. 2. Data transmission: as the GPU device memory is limited and data transmission overhead, passed directly to the GPU routing tables end

processing equipment, cost is relatively expensive, so the only purpose of extracting sorted routing table all network address and subnet mask categories, respectively transmitted to open up the device end of the memory space corresponding position in advance. The IP entry is passed to the device, and the host end is completed. In the Post stage, the final result is handled, and the processing of the package is forwarded or discarded. The most important is the GPU phase, which is mainly responsible for routing retrieval and return of results. Then the implementation of two different routing retrieval algorithms on GPU architecture is introduced.

B. The Implementation of CUCKOO Hash on GPU

In the GPU implementation phase, parallel retrieval is mainly performed: the longest matching principle of the subnet mask is used for route retrieval. When multiple entries in a routing table in the IP network can match the destination IP, the one with the longest mask is generally used as the matching item and determines the next hop. Therefore, in the concrete implementation retrieval, need to put the IP entry with each subnet mask in the routing table operations, again in the routing table lookup matching network, select the longest entry in subnet mask for the match, as shown in figure 3.2 the realization of the CUCKOO hash on the GPU.

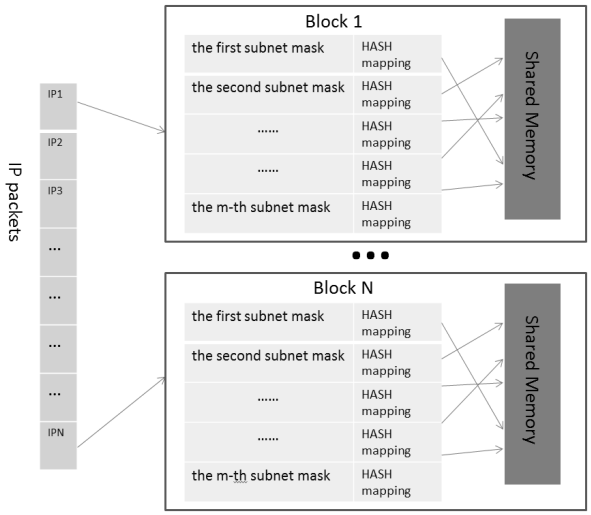

Figure 3.2 The implementation of CUCKOO Hash on GPU

The specific solutions On the GPU parallel implementation : 1. The use of GPU parallel features, handle multiple IP to find the items at the same time, for each of the thesis& IP address to open a Block, the Block, the number of threads in the number of subnet mask in the routing table. For example, with 32,30,24,22,20,16,8 subnet mask in the routing table, a total of 7 seed nets mask, then the distribution in each Block 7 threads, each thread with a subnet mask and IP operation, and then through the CUCKOO hash map to the routing table, to detect whether there is the current address after the operation, find corresponding to the result of the query to the routing entry in posterior from the location in the table, 7 thread cooperation to check an address, due to the routing table has been ordered by the length of the subnet mask, seven query to the thread route matching the location of the item, the top is expressed as the longest prefix match. Write back the results of the query to the CPU. In this way, multiple IP address lookups are processed, and one IP address simultaneously computes and finds the different subnet masks, speeding up the whole search process. The method is used for routing and searching for 400 million IP addresses per second

C. The Implementation of LCtrie on GPU

By using LCtrie routing lookup, find multiple bits in each step of the inspection address, find the step width of K Trie tree branches, the maximum number of branches of each node is 2 ^ K. Use the longest prefix matching lookup to find the nearest left endpoint in the left endpoint set. Because the implementation of LCTrie algorithm requires frequent I/O operations, I/O operation is a typical long delay operation, and the means of hiding the delay of the visit include: 1. Multi-threading technology is the main means of hiding long delay. 2. I/O vectorization: the contents of the continuous storage can be completed by an I/O instruction. 3. I/O stagger: the non-dependent instructions are arranged near the sub-instruction or the thread switch instructions, filling in the delay slots produced by other instructions. However, the above methods are CPU based, and can only reduce the partial delay, and the overall efficiency of the algorithm is not improved well. In this paper, GPU parallel technology is proposed to use GPU's large number of threads and the advantages of processing massive data, so as to hide the delay by improving the total data Throughput (Throughput).

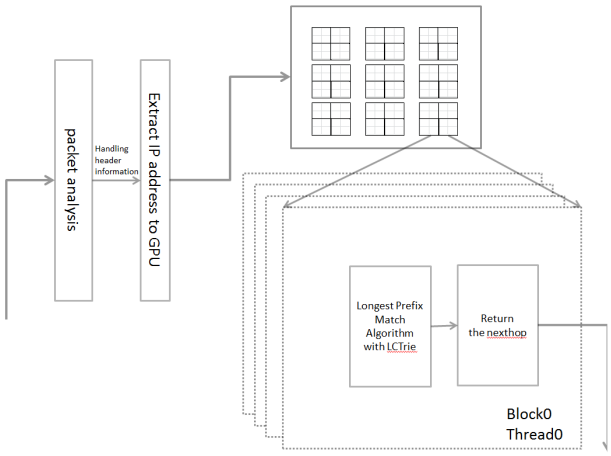

Figure 3.3 GPU query framework

As shown in figure 3.3 for LCTrie queries in parallel to the GPU, the framework of heterogeneous parallel characteristic of using the GPU, first on the CPU for data preparation, transmit data to the GPU devices, parallel computing, using multiple SP (streaming processor basic processing unit) for processing at the same time. Multiple SP plus other resources make up a SM (streaming multiprocessor. Other resources are storage resources, Shared memory, and storage etc.). Open multiple threads, 32 threads are one warp, one or more warp partitions to a block. Every time the GPU scheduling a warp in the 32 threads to execute the same instruction, the individual threads corresponding to different data resources, so each thread can get an IP, to LCTrie retrieval, and returns the nexthop. When assigning blocks, the reasonable number of blocks should be allocated according to the number of SM of GPU, so that the SM of GPU can be utilized to improve the utilization rate.

The accelerated study of LCTrie algorithm in parallel to GPU has been completed. Experimental results show that the efficiency of LCTrie query algorithm is increased by 30 times. Therefore, the algorithm has parallelism, which is suitable for parallel transplantation, thus improving the performance of look-up, and can adapt to the greater environmental demand of flow. The implementation of the algorithm on GPU has simple operation, high performance and high cost performance, which is suitable for high-speed forwarding software design.

IV. EXPERIMENT

A. The Performance of IP lookup

The main hardware of the experiment: Intel CPU: i7-6700 CPU. NVIDIA GPU: GTX750Ti. GTX750Ti has 640 CUDA core, core frequency 1110/1189mhz and memory type GDDR5 capacity 2GB. Experimental platform: Linux system version is Ubuntu 14.04. The data source of the test is Nilsson. S and Karlsson. G is using the test data set

used in the LCtrie performance experiment [14], which is derived from the FUNET and is the actual data of the data collected. The experiment tested the comparison between CPU and speed of the LCtrie algorithm on the CPU and the GPU. This test shows the performance of IP routing lookup tasks based on the kernel level GPU computing API. Using the routing table of 1000 routing entries, the test shows that the execution speed of the algorithm is shown in table 4.1.1. When the number of packets is increased, the efficiency of parallel acceleration is higher, which is consistent with the characteristics of GPU to speed up the multi-data set, which proves the feasibility of GPU acceleration route retrieval.

TABLE 4.1.1

THE COMPARISON OF SPEED OF LCTRIE ALGORITHM RUNNING ON CPU AND GPU

	10k	20k	30k	40k	50k	60k	70k	80k	90k
CPU(us)	192.90	407.46	572.26	727.97	907.25	1158.11	1297.10	1538.05	1699.62
GPU(us)	25.02	28.95	32.51	35.54	38.39	42.48	43.86	49.41	52.74
speed up	7.71	14.07	17.60	20.48	23.63	27.26	29.57	31.13	32.23

IP routing lookup is the kernel application that uses GPU acceleration in this paper, and there are two kinds of algorithms: CUCKOO FILTER and LCTrie. The experiment compares these two algorithms. In this paper, the comparison between the two algorithms is compared with the comparison between the two algorithms. Figure 4.1 CUCKOO FILTER and LCTrie are used to retrieve time-consuming comparisons. Because of the randomness of time of thread scheduling, in order to obtain a more balanced result, it is time to calculate the average time of 50 retrievals during the test. The result of the experiment was: when processing 10,000 packets, the Hash retrieval time was 40.90 us, and the LCtrie retrieval time was 23.15 us. When processing 100,000 packets, the Hash retrieval takes 252.02 us, and the LCtrie retrieval takes 45.80 us. As the graph shows, the larger the number of packets, the more time it takes to use CUCKOO hashes, which means that the LCtrie retrieval efficiency based on GPU architecture is higher than that of CUCKOO hashes when processing a large number of packet retrieval.

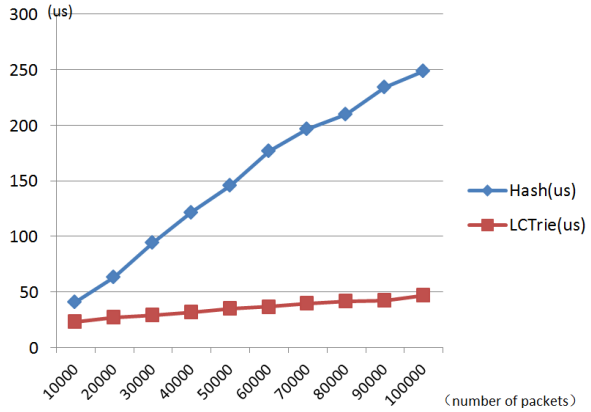

Figure 4.1 the time comparison between CUCKOO FILTER and LCTrie

The following three forms are small table (100 items, table 4.1.2), central table (1,000 items, table 4.1.3), and large table (10000 items, table 4.1.4). Each routing table has nine sets of data for query routing table operations, and the two methods for each group of data are tested on average time spent on each of the 50 queries.

TABLE 4.1.2

THE ROUTING TABLE ITEM HAS 100 ENTRIES

	10k	20k	30k	40k	50k	60k	70k	80k	90k	100k
Hash(GPU)	40.90	63.16	94.28	121.70	145.98	176.52	196.62	209.88	234.02	252.02
LCTrie(GPU)	23.15	26.77	28.84	31.66	34.70	36.55	39.53	41.76	43.35	45.80

LC-Trie(CPU)	146.17	257.07	385.85	784.50	827.28	882.54	901.32	937.51	1049.15	1400.21
Speedup (CPU and GPU)	6.31	9.60	13.38	24.78	23.84	24.15	22.80	22.45	24.20	30.57
Speedup (Hash and LC)	1.77	2.36	3.27	3.84	4.21	4.83	4.97	5.03	5.40	5.50

TABLE 4.1.3

THE ROUTING TABLE ITEM HAS 1000 ENTRIES

	10k	20k	30k	40k	50k	60k	70k	80k	90k	100k
Hash(GPU)	65.16	113.82	167.52	214.18	278.93	313.72	360.86	411.08	440.92	476.44
LCTrie(GPU)	25.02	28.95	32.51	35.54	38.39	42.48	43.86	49.41	52.74	55.74
LC-Trie(CPU)	192.90	407.46	572.26	727.97	907.25	1158.11	1297.10	1538.05	1699.62	2190.09
Speedup (CPU and GPU)	7.71	14.07	17.60	20.48	23.63	27.26	29.57	31.13	32.23	39.29
Speedup (Hash and LC)	2.60	3.93	5.15	6.03	7.27	7.39	8.23	8.32	8.36	8.55

TABLE 4.1.4

THE ROUTING TABLE ITEM HAS 1000 ENTRIES

	10k	20k	30k	40k	50k	60k	70k	80k	90k	100k
Hash(GPU)	92.90	160.24	234.06	302.80	350.18	399.63	514.84	520.74	569.06	625.18
LCTrie(GPU)	27.14	31.09	36.20	40.74	44.47	47.78	52.08	56.71	61.11	65.27
LC-Trie(CPU)	200.73	437.75	606.69	783.58	956.17	1241.73	1392.83	1554.92	1753.51	2288.85
Speedup (CPU and GPU)	7.40	14.08	16.76	19.23	21.50	25.99	26.74	27.42	28.69	35.07
Speedup (Hash and LC)	3.42	5.15	6.47	7.43	7.87	8.36	9.89	9.18	9.31	9.58

B. Analysis of experimental result

According to the experimental data graph, LCTrie is superior to CUCKOO FILTER based on the GPU architecture accelerated routing retrieval method, and the larger the routing table, the more data packets are retrieved, the more obvious the acceleration of LCTrie is. CUCKOO FILTER is a hash lookup, which USES hash algorithm to retrieve the routing, which is simple, and the hashing search efficiency is O (1). However, there are two more prominent problems with hash lookup for routing retrieval: the hash conflict and the longest prefix matching hash for efficiency problems. This paper has taken some measures to solve these problems, but the effect is still not satisfactory. Analysis of causes:

1. The hash algorithm has a hash conflict. It is known from the experimental data that when the routing table is larger, the retrieval efficiency is lower, and the collision occurs when the hash map of the routing table is established. In this paper, CUCKOO hashing is selected to solve some of the hash conflicts by using two hash buckets in the algorithm, but the problem of conflict resolution still exists. A specific hash function is only suitable for a certain number of matches, it's almost impossible to find a common hash function, can adapt to from a few matches to tens of millions of a match situation, in general, with the increase of the match, hash collision with will also increase, and its time complexity is not controlled, it is a big problem, this problem prevents the routing lookup hash algorithm to the core routers

2. A hash retrieval rate has been below LCtrie retrieval, the reason is that when retrieved routing, must carry on the longest prefix match, when using the hash algorithm to retrieve, the worst case needs 32 hash lookup. Although hash is a O (1) the

lookup, efficiency is very high, however, due to the routing lookup USES the longest prefix match the way to find the next-hop exports, therefore, when using hash lookup algorithm, is based on the traversal of subnet mask to achieve the longest prefix match, that is to say, if one will eventually pass from the default gateway IP datagram, the worst cases need to match the 32 times to get a result, the efficiency of routing lookup is not one hash lookup the efficiency of the O (1). In this article, on the transfer of the algorithm to the GPU, using GPU lots of threads, each 32 threads for 1 set, each thread with a subnet mask operation after the hash retrieval, this ensures a 32 subnet mask matching at the same time. This solves the problem of multiple matching of a message and can handle multiple messages simultaneously to make up for the defect of hash algorithm and route retrieval. But due to the nature of the GPU thread cooperation, cooperation in multiple threads to find only one IP address, despite the implementation of the parallel different threads at the same time according to different subnet mask after operation, the route table lookup, can be in one table, get all the matching results. However, due to the final selection of the longest prefix matching, the synchronization of threads is required on GPU and the exchange of data between multiple threads is implemented. The data exchange here refers to the selection of the longest routing matching entries in all matches, which can cost a certain amount of time.

By using Trie tree to retrieve the routing is an ordered tree that is a combination of the direct addressing table and the tree. Prefix Trie features are: a string of public information, such as string "pine" and "pineapple" public prefix is "pine", when the match string "pineapple", must first be discovered the existence of a "pine", so minimize unnecessary string comparison, the query efficiency is higher than a hash table. LCTrie is a combination of path compression and multiple trie nodes to further optimize the basic trie structure by increasing the number of bits per lookup (step length) to achieve quick lookups. In this paper, the LCtrie algorithm is transplanted to GPU, which utilizes the characteristics of a large number of threads in GPU to make routing query more efficient.

V. CONCLUSIONS

The rapid growth of network speed has put great pressure on the existing routing system. At present, the transmission speed of the physical layer is increasing, and the message processing capacity of the routing system becomes the bottleneck of limiting data traffic growth. Most of the messages in current network traffic are still in IPV4 format, and the capability of IPV4 message processing is improved to the current research focus. The industry has proposed various solutions such as hardware acceleration, algorithm optimization, multi-core and multi-threading. However, these schemes still have high cost and low efficiency. Because the multi-core processor and GPU are already very cheap, and the code optimization, to quantization and parallelism have penetrated into the bone marrow of the IT industry, the algorithm is optimized by using the characteristics of GPU. This paper analyzes the existing routing table lookup algorithm research, introducing the CUCKOO hash, a low collision rate of hash algorithm, using LCTrie level compression tree algorithm, on the two kinds of algorithm was transplanted to the GPU to do parallel acceleration, especially routed LCTrie algorithm based on GPU retrieval, achieved good effect, will help to further research work.

REFERENCES

[1] Gupta, P., Lin, S., & Mckeown, N. (1998). Routing lookups in hardware at memory access speeds. Proc of Infocom', 3, 1240-1247 vol.3.

[2] Liu, H. (2001). Routing prefix caching in network processor design. Computer Communications and Networks, 2001. Proceedings. Tenth International Conference on (pp.18-23). IEEE.

[3] Lu, W., & Sahni, S. (2010). Low power tcams for very large forwarding tables. IEEE/ACM Transactions on Networking, 18(18), 316-320.

[4] Kritikos, W. V., Rajasekhar, Y., Schmidt, A. G., & Sass, R. (2011). A radix tree router for scalable fpga networks. IEEE, 76-81.

[5] Lampson, B., Srinivasan, V., & Varghese, G. (1999). Ip lookups using multiway and multicolumn search. Networking IEEE/ACM Transactions on, 7(3), 324-334.

[6] 刘亚林, 刘东, & 张晓. (2001). 基于前缀扩展的快速路由查找算法. 计算机学报, 24(12), 1272-1278.

[7] 张怀庆. (2008). 基于 TCAM 和多核处理器的高速路由查找转发引擎设计. (Doctoral dissertation, 山东大学).

[8] Quan, W., Xu, C., Guan, J., & Zhang, H. (2014). Scalable name lookup with adaptive prefix bloom filter for named data networking. Communications Letters IEEE, 18(1), 102-105.

[9] Särelä, M., Näslund, M., & Nikander, P. (2012). PACKET ROUTING IN A NETWORK BY MODIFYING IN-PACKET BLOOM FILTER. US, US 20120287934 A1.

[10] Fan, B., Kaminsky, M., & Andersen, D. G. (2013). Cuckoo filter : better than bloom. ;login:: the magazine of USENIX & SAGE, 38, págs. 36-40.

[11] Natarajan, A., & Subramanian, S. (2012). Bloom filter optimization using Cuckoo Search. International Conference on Computer Communication and Informatics (pp.1 - 5). IEEE.

[12] Ravikumar, V. C., Mahapatra, R., & Liu, J. C. (2002). Modified lc-trie based efficient routing lookup. 177-182.

[13] Pao, D., & Li, Y. K. (2003). Enabling incremental updates to lc-trie for efficient management of ip forwarding tables. IEEE Communications Letters, 7(5), 245-247.

[14] Stefan Nilsson and Gunnar Karlsson. IP-address lookup using an LC-trie. IEEE Journal on Selected Areas in Communications,1999,17(6):1083-1092