Fast Update of Forwarding Tables in Internet Router Using AS Numbers *

Heonsoo Lee, Seokjae Ha, and Yanghee Choi
School of Computer Science and Engineering
Seoul National University
San 56-1, Shilim-dong, Kwanak-ku, Seoul, Korea
{grace, sjha, yhchoi}@mmlab.snu.ac.kr

Abstract

The updates of router forwarding tables can be made faster using the Autonomous System number corresponding to a prefix as an intermediate number between the prefix and the next-hop address. At the cost of fast update, one table lookup introduces small additional delay, which can be eliminated by pipelining. This scheme is applicable to several routing table lookup algorithms for fast update.

1. Introduction

The IP address lookup is a process finding a next-hop address from a destination IP address. This process is so important function of routers that it must be fast, and implemented with a reasonable cost. Before CIDR(Classless Inter-Domain Routing) [2], the lookup could be done easily with comparisons of fixed length prefixes. The shortage of 32bit IPv4 addresses, especially the shortage of class-B addresses, stems from the fact that there is no proper address class for medium-sized organizations. A class-C address, which can accommodate up to 254 hosts, is too small, and a class-B address, which can support up to 65534, is too big for them. The growing size of the routing tables in the Internet routers was also a problem [2]. With the introducing of CIDR in 1993, new network addresses are allocated, and the increasing rate of routing table size is slowed down. However it also introduced the Longest Prefix Matching(LPM) problem in the routing lookups. Here, 'prefix' means a set of networks. A prefix is of variable length, and the longer a prefix is, the more specific routing information it has. LPM compares a destination IP address and each prefix in the forwarding table and produces a set of matching prefixes. Among them, the longest prefix, which has the most specific routing information, is selected. Because this direct method is inefficient, many faster schemes have been suggested [1, 4, 5, 6, 11, 12, 13, 14]. Most uses more memory to reduce the time.

Routers can be classified into three categories [3]. Routers in access networks allow homes and small businesses to connect to an ISP(Internet Service Provider). Routers in enterprise network link computers within a campus or enterprise. Routers in the backbone link together ISPs and enterprise networks with long distance trunks. It is the backbone routers that have difficulties in LPM, because of the huge routing table size. This paper is focused on backbone routers.

Many schemes concentrate on fast lookup time at the cost of slow update. Because of frequent routing table update of the Internet, however, it is important to invent update method applicable to many schemes for fast update. We propose such a method that does not increase memory or time complexity of lookup. As an instance, we show that this method improves [1].

This paper is organized as follows. Section 2 describes previous works, and section 3 presents our proposed scheme. Section 4 discusses impact of routing instability to forwarding table update schemes.

2. Previous work

The major performance bottleneck in backbone IP routers is the time taken to lookup a route in the forwarding table [3]. Because memory access is much slower than code execution, the lookup time depends primarily on the memory access time. In other words, the speed of a route lookup algorithm is determined by the number of memory accesses to find the matching route entry, and by the memory speed. By the memory size used, lookup schemes can be classified

*This work was supported in part by the Brain Korea 21 project of Ministry of Education, in part by the National Research Laboratory project of Ministry of Science and Technology, and in part by Electronics and Telecommunications Research Institute, 2000, Korea.

into two categories.

The first category uses high-speed memory to reduce time. Generally, memory size is reduced by compressing the forwarding table into a small high-speed memory at the cost of increased number of memory accesses. Because the meaningful prefixes exist very sparsely in the prefix space of 2^{32}, efficient forwarding table compaction can be achieved. As memory prices drop, however, this may be the wrong design decision [3].

Lampson et al. [4] suggested binary search on the sorted IP address pairs, where a pair corresponds to maximum and minimum value of IP addresses for a prefix. Very small memory is used, but sorting makes the update difficult. And the binary search takes too long in case of large table. So this is applicable only to medium-sized enterprise routers.

Waldvogel et al. [5] suggested prefix grouping according to length. From the longest prefix group, it tries to find the matching prefix. In a group, the wanted one can be found in a memory access by perfect hashing. Binary search, instead of linear one, can find the longest prefix in log L times among L groups. This scheme is scalable for address lengths, and is applicable to IPv6. Because of perfect hashing, however, the entire forwarding table must be regenerated at every update. And the lookup time is not fixed, and it exhibits long worst case time.

Degermark et al. [6] suggested a very efficient indexing scheme. It assumes a complete binary tree containing prefixes. And the tree is cut into three levels at heights 16, 24 and 32. At the first level, a bit vector of 2^{16} bits represents 2^{16} nodes, and a node with a set bit at the bit vector has a pointer to a next-hop address or a next-level trie. The second category aims to reduce the number of memory accesses. The extreme method uses a table of 2^{32}(4G) entries of IP address, and the corresponding next-hop address. Then, lookup can be done in one memory access. However, 4GB memory (assuming next-hop is denoted in 8bit) is still expensive and it has difficulties in initializing and updating. However, required memory size can be significantly shrunk if the table is divided into several smaller ones.

Gupta et al. [1] suggested two tables, one is for prefixes shorter than or equal to 24bit, and the other for prefixes longer than 24bit. As observed in the present Internet prefixes shorter than or equal to 24bit amounts to 99.93%. The big table (TBL24) is for short prefixes (2^{24}entires), and the small table (TBLlong) for a few long prefixes. The TBL24 has next-hop addresses for normal prefixes, but has pointers into the TBLlong in case of long prefixes. In practice, the required memory can be reduced to 33MB(TBL24(32MB)+TBLlong(1MB)). Separating the two tables into independent memory banks enables pipelining (Fig. 1). Therefore, one lookup can be done in one memory access time.

The major drawback of this scheme is that many table

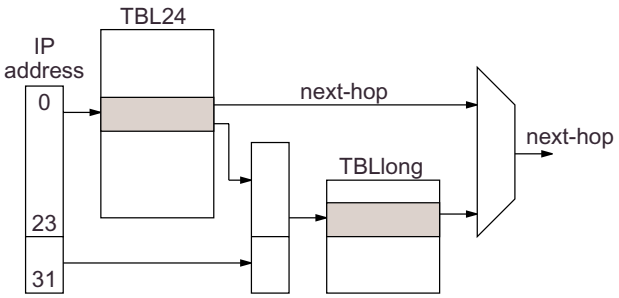


Figure 1. Pipeline architecture. The next hop is directly obtained.

entries need to be changed to update a prefix. Each prefix must be expanded to 24bit regardless of its original length, so an 8bit prefix, for example, is mapped to 65536 entries. Therefore, to change the next-hop address of the 8bit prefix, as many as 65536 entries must be changed. Generally, for a prefix of length k, the update time complexity is $O(2^{24-k})$. The solutions suggested by [1] do not reduce the overall burden of updates, but only move the burden from processor to a specially designed hardware. As before, lookup cannot proceed during updates, and each update needs burst memory accesses.

Recently, many newly-suggested lookup schemes adopt advantages of both categories. The first 16bits of IP address are used as in the first category and the remnant bits are used as in the second category.

3. The proposed scheme

3.1. Update

The aim of this scheme is to reduce the update overhead of forwarding table. For that purpose, a stage of indirection is introduced between prefixes and next-hop addresses. In spite of frequent changes of next-hop addresses, the change should be restricted between intermediate number and next-hop addresses, and the change itself should be easy. The intermediate number should have following features.

- The intermediate number itself should not change frequently in order to retain the relation between prefixes and next-hop addresses.
- Every prefix should have only one intermediate number.
- It should be short.
- It should be easily obtained in backbone routers. It would be unreasonable to prepare additional protocols only to get the intermediate number.

We decided to use Autonomous System (AS) number for the destination IP address as the intermediate number. The Internet can be modeled as interconnected ASes and backbone routers are located at AS boundary [18]. AS advertises prefixes under its control with its 16bit AS number through BGP(Border Gateway Protocol) [7]. The origin AS number, to which a prefix belong, can be obtained from AS-PATH in the routing table. It can be downloaded from the routing table, and satisfies the above requirements as intermediate number.

Therefore, the AS number, instead of next-hop address, is looked up first, and then the next-hop address. Assuming that next-hop addresses of a router are fewer than 256, 64KB of memory is required because AS number is 16bits.

AS(16bit)	Next-hop(8bit)
0	A
1	A
2	B
...	...
65534	C
65535	B

Table 1. AS2NH(AS to Next Hop address) Table

The next-hop address can be retrieved in one memory access, and updated in one memory access in case of an AS-PATH change(Table 1). If the destination is the router's own AS, packets should be forwarded differently according to its prefix. To assign different next-hop for each local prefix, we suggest the allocation of the upper area of AS numbers. As in Table 2, the bottom area is used for globally allocated AS numbers, which are as many as 16971 at the time of this writing(July 2000), and the top area is reserved for private use in AS by IANA [8]. We reserve the top 1024 AS numbers for long prefixes (i.e. it is assumed that a backbone router does not have more than 1024 entries for prefixes longer than 24bits). And for prefixes for local AS, the area between 16972 and 64511 can be allocated from the top (i.e. 64511).

Because an additional stage is introduced (AS to NH), entire lookup delay is increased. However, the delay can be reduced if the 64KB of table is made by high-speed memory. Because of its simplicity, embedding this scheme into existing lookup schemes is trivial.

3.2. Lookup

The proposed scheme changes the original two-table scheme [1] (Fig. 1) to three-table one (Fig. 2) where AS2NH table is inserted into the pipeline as the last stage.

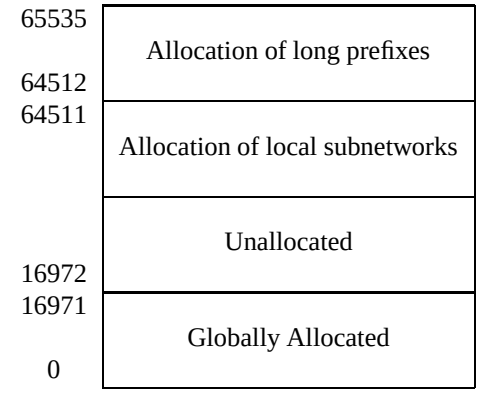


Table 2. An allocation map of AS numbers.(July 2000)

The three tables are as follows.

- TBL24 : With 2^{24} address space and each entry of 16bit AS number, the memory amounts to 32MB.
- TBLlong : This table can accommodate up to 1024 prefixes, where each prefix has 256 sub entries. Each entry with 16bit AS number, the required memory amounts to 512KB (1024*256*16bit).
- AS2NH : With 2^{16} address space and each entry of 8bit next-hop address, the memory amounts to 64KB.

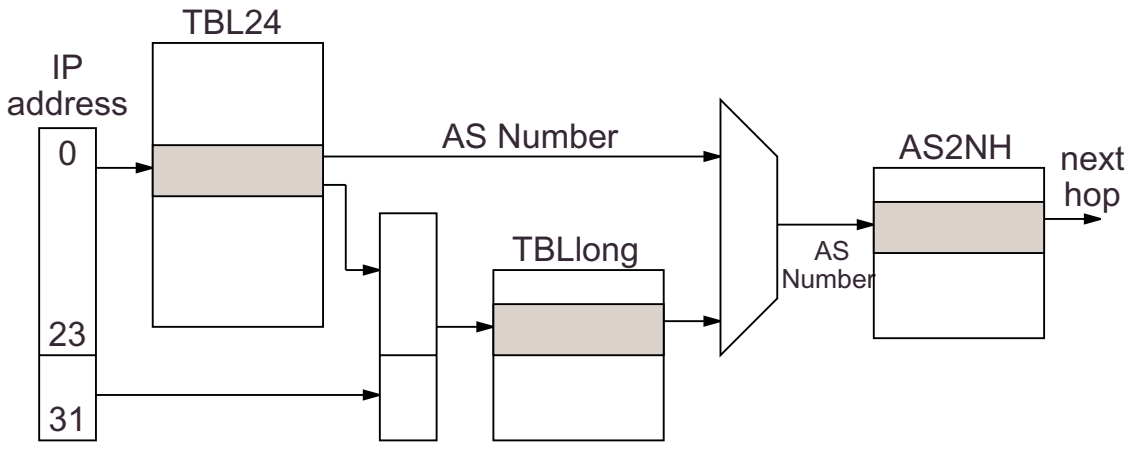
Among AS numbers, the top most 1024 numbers(64512∼65535) are reserved for private use of AS. We use these numbers as indicators of long prefixes in TBL24. If we encounter a number smaller than 64512 in TBL24, that corresponds to normal AS number. Otherwise, that corresponds to long prefix, so TBLlong is referred to find out the AS number. The numbers greater than 64512(1111110000000000_2) can be easily found out by checking the most significant six bits. The remaining ten bits point to the location of entry in TBLlong.

As in the original scheme(Fig. 1), the separation of three tables into independent three memory banks enables pipelining. And although increased to three stages, the hardware pipeline sustains the maximum throughput(Fig. 2).

3.3. Examples

Assuming two prefixes, 147.46/16, 147.46.114.128/28, AS numbers 9488, 2563, and next-hop addresses A, B, respectively, the three tables are as follows(Fig. 3).

If the destination IP address is 147.46.115.31, then TBL24 returns 9488, and then AS2NH returns A as the next-hop address.



h

Figure 2. Proposed revised pipeline architecture. AS number is converted to next-hop.

For 147.46.114.83, which belongs to a long prefix, TBL24 returns 64555. Because the value is over 64512, it indicates that the IP address belongs to long prefix. The difference 43(64555 - 64512) corresponds to the entry number in TBLlong. Then, the number is multiplied by 256 and added by the least significant 8bits of IP address(83), resulting in the corresponding sub entry number(43*256+83). Then, AS2NH returns B as the next-hop address for AS2563.

If the next-hop address of AS9488 changes to C, only the corresponding entry in AS2NH table is changed with one memory access.

4. Routing instability

4.1. Ideal situation

In an optimal stable wide-area network, routers should only generate routing updates for relatively infrequent policy changes and the addition of new physical networks [9]. Insertion or deletion of a prefix(or a physical network) result in burst memory access to change AS numbers of TBL24 or TBLlong. Table 3 represents the comparison of two snapshots taken from the global prefix databases, which have 7-day-gap between them [17]. About 1000 prefixes are changed, which corresponds to 1 in every 10 minutes.

This amount of changes does not matter much. And the other kind of updates, the change of next-hop address, can be reflected into AS2NH in one memory access. Therefore, the suggested scheme seems to be suitable for the stable Internet.

Time of snapshots	Deleted prefix	Inserted prefix
1/10/97, 1/17/97	571	850
10/17/97, 10/24/97	645	549
7/8/00, 7/15/00	288	594
8/3/00, 8/10/00	338	547

Table 3. Examples of changes in snapshot of prefixes.

4.2. Route flaps

But, the real Internet routing has been known to be unstable [9]. The instability comes mainly from exchanging large amounts of redundant update messages among backbone routers. A prefix is advertised as having been withdrawn, which is not, and a few minutes later, is announced as appeared. Such update messages sometimes amounts to a few hundreds in a second.

There are three types of routing updates: forwarding instability(the network has to choose a new path), policy changes(BGP is announcing something new that does not really affect routing), redundant information(junk). Especially, among the redundant information, the duplicate withdraw announcements is predominant(more than 99% of BGP information [9].)

Among the proposed solutions for this instability, dampening [10] is to mark unstable routes, and delay update information for those, because the update messages are concentrated on those unstable ones. However, delaying every update message is not suitable for really changed routes, so immediate update for normal routes should be guaranteed. Another method is to neglect update messages for prefixes

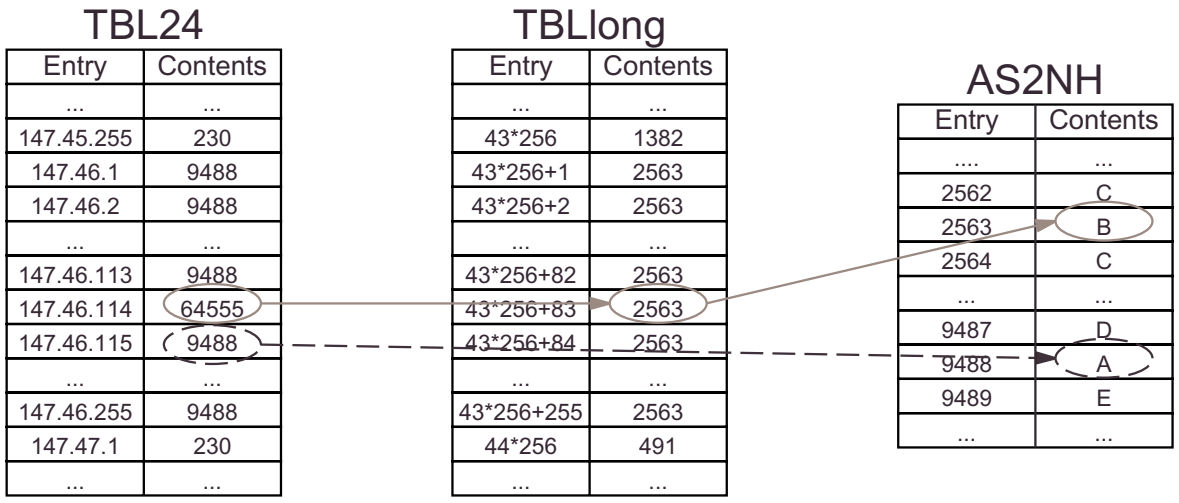


Figure 3. Example of TBL24, TBLlong, and AS2NH.

longer than a threshold, for longer prefixes tend to be unstable. It is also helpful to refuse to peer with small ISPs, for their usual instability.

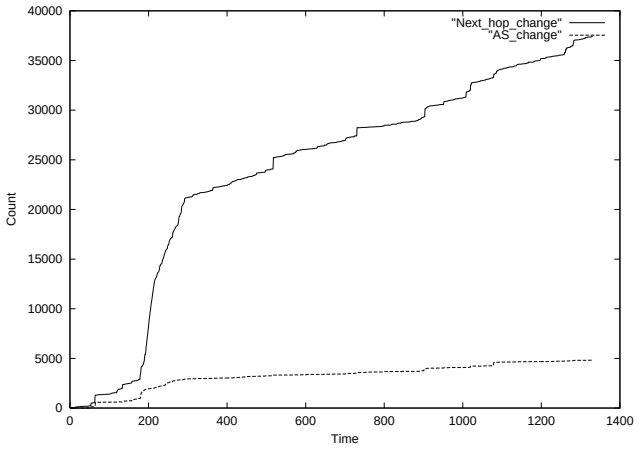


Figure 4. Cumulative frequency graph of AS changes and next hop changes for 14 days.

5. Simulation result

We want to show that how much the proposed algorithm reduces memory accesses as respect to previous ones. We regard [1] as a representative of previous ones, so used it as the reference. [1] requires many memory accesses for every next hop update, and the proposed algorithm requires about the same amount of memory accesses for every AS update, but only 1 access for next hop update. Therefore, we measured how often AS and next hop changes in a real backbone router. And, we eliminated some route flaps before simulation by accepting only explicitly different ASes or next hops compared to previous ones.

Date	10/2	10/5	10/8	10/12	10/15
AS	599	3023	3422	4667	4816
Next hop	9022	22377	26793	34706	37496

Table 4. Cumulative frequency table of AS changes and next hop changes for 14 days.

Routing table snapshots and BGP update messages of an NAP are available through [19]. Routing table snapshots are provided as HTML format, and have over 40,000 default-free prefixes, each of which may have multiple path information. BGP update messages are provided as binary raw data through FTP, and are aggregated into a file for every 15 minutes. We used all the data from 10/1/2000 21:45 to 10/15/2000 17:45 for simulation. First, we constructed a routing table by parsing a routing table snapshot. Then, we updated the routing table with decoded BGP update messages. Because of bursty characteristics of update, we show the result as cumulative frequency graph(Fig. 4) and table(Table 4).

In spite of abnormal periods of highly frequent updates, next hop changes 7 times more frequently than AS in average. Because the intermediate number should not change frequently in order to retain the relation between prefixes

and next-hop addresses, we can conclude that AS number is suitable for that purpose.

6. Conclusion

In this paper, we propose new scheme that reduces update overhead in the Internet forwarding table. Some lookup algorithms have large update overhead due to prefix expansion. We suggest using the AS number of prefix as intermediate number between IP and next-hop address for fast update. As the simulation result from real BGP update messages, AS number changes 7 times less frequently than next hop. This method can be easily applied to other lookup schemes just by inserting a small table that converts AS to next-hop.

References

[1] P. Gupta, S. Lin and N. McKeown, *"Routing Lookups in Hardware at Memory Speeds,"* INFOCOM, pp. 1240-1247, 1998.

[2] V. Fuller, T. Li, J. Yu and K. Varadhan, *"Classless Inter-Domain Routing (CIDR) : an Address Assignment and Aggregation Strategy,"* RFC 1519, IETF, Sept. 1993.

[3] S. Keshav and Rosen Sharma, *"Issues and Trends in Router Design,"* IEEE Communications Magazine, May 1998.

[4] B. Lampson, V. Srinivasan and G. Varghese, *"IP Lookups Using Multiway and Multicolumn Search,"* INFOCOM, pp. 1247-1256, 1998.

[5] M. Waldvogel, G. Varghese, J. Turner and B. Plattner, *"Scalable High Speed IP Routing Lookups,"* SIGCOMM, pp. 25-35, 1997.

[6] Mikael Degermark, A. Brodnik., S. Carlsson and S. Pink, *"Small Forwarding Tables for Fast Routing Lookups,"* SIGCOMM, pp. 3-14, 1997.

[7] Y. Rekhter and T. Li, *"A Border Gateway Protocol 4 (BGP-4),"* RFC 1771, IETF, Mar. 1995.

[8] J. Hawkinson and T. Bates, *"Guidelines for creation, selection, and registration of an Autonomous System (AS),"* RFC 1930, IETF, Mar. 1996.

[9] C. Labovitz, G. Robert Malan and F. Jahanian, *"Internet Routing Instability,"* IEEE/ACM Transactions on Networking, vol. 6, no. 5, pp. 515-528, Oct. 1998.

[10] C. Villiamizar, R. Chandra and R. Govindan, *"BGP Route Flap Damping,"* RFC 2439, IETF, Nov. 1998.

[11] V. Srinivasan and G. Varghese, "Fast address lookups using controlled prefix expansion," ACM Sigmetrics, 1998.

[12] S. Nilsson and G. Karlsson, "IP-Address Lookup Using LC-Tries," IEEE JSAC, vol.17, no.6, pp. 1083-1092, June 1999.

[13] H.H.-Y. Tzeng and T. Przygienda "On Fast Address-Lookup Algorithms," IEEE JSAC, vol.17, no.6, pp. 1067-1082, June 1999.

[14] N. F. Huang and S. M. Zhao, "A Novel IP-Routing Lookup Scheme and Hardware Architecture for Multigigabit Switching Routers," IEEE JSAC, vol.17, no.6, pp. 1093-1104, June 1999.

[15] V. Srinivasan, S. Suri and G. Varghere, "Packet Classification Using Tuple Space Search," SIGCOMM, pp. 135-146, 1999.

[16] P. Gupta and N. McKeown, "Packet Classification on Muliple Fields," SIGCOMM, pp. 147-160, 1999.

[17] The Cidr Report, http://www.employees.org/ tbates/cidr-report.html.

[18] Y. Rekhter and P. Gross, "Application of the Border Gateway Protocol in the Internet," RFC 1772, IETF, Mar. 1995.

[19] The Internet Performance Measurement and Analysis (IPMA) project, http://www.merit.edu/ipma/