A fresh look at Forwarding Information Base compression via mathematical analysis

Conference Paper · May 2014

DOI: 10.1109/NOMS.2014.6838342

Tong Yang
Peking University

Kave Salamatian
Université Savoie Mont Blanc

A Fresh Look at Forwarding Information Base Compression via Mathematical Analysis

Tong Yang[1,2], Gaogang Xie[1], Kavé Salamatian[3]
[1]Institute of Computing Technology, Chinese Academy of Sciences (CAS), China.
[2]DNSLAB, China Internet Network Information Center, Beijing 100190. [3]University of Savoie, France.

***Abstract*[1]—With the fast development of Internet, the size of routing table in the backbone router continues to grow rapidly. Forwarding Information Base (FIB), which is derived from routing table, is stored in line-card to conduct routing lookup. Since the line-card's memory is limited, it would be worthwhile to compress the FIB for consuming less storage. Therefore, various FIB compression algorithms are proposed [2-7]. However, there is no well-presented mathematical support for the feasibility of the FIB compression solution, nor any mathematical derivation to prove the correctness of these algorithms. To address these problems, we propose a universal mathematical method based on the *Group*[2] theory. By defining a *Group* representing the Longest Prefix Matching Rule (LPM), the bound of the worst case of FIB compression solution can be figured out. Furthermore, in order to guarantee the *ultimate correctness* of FIB compression algorithms, Routing Table Equation Test (RTET) is proposed and implemented to verify the equivalence of the two routing tables before and after compression by traversing the 32-bit IP address space.**

I. Introduction

The backbone routing table has been growing at an exponential rate, driven mainly by multi-homing and the rapid development of mobile communication [1]. The fast increasing routing table incurs fast increasing FIB. For the routing lookup schemes based on software [8-10], FIB compression can be used to reduce their memory requirements; for the routing lookup algorithms based on TCAM [11-13], FIB compression can be used to reduce the hardware cost and power consumption. Therefore, a variety of FIB compression algorithms are proposed [2-7]. These algorithms compress the routing table by transforming the binary trie structure.

In addition, the routing tables' prefixes are overlapped, which means that some prefixes are a part of others. This brings many negative effects on the performance of routing lookup and incremental update [15]. There are mainly two overlap elimination algorithms: Leaf-pushing [14] and ONRTC [15] algorithm. They can totally eliminate the overlap also by transforming the binary trie[3].

However, is FIB compression solution feasible? What's the worst case of the FIB compression solution? How to guarantee the correctness of trie-transformation algorithm? Current FIB compression algorithms just compress the routing table, regardless of the size and structure of the routing table. In contrast, the feasibility, effectiveness and correctness of FIB compression algorithms are emphasized and well-studied in this paper.

a) Feasibility and effectiveness. According to the information theory, it is definite that the compressed routing table holds the information equivalent to the original one. Therefore, if and only if there is redundancy in the original routing table, the FIB compression solution is feasible. Then is there redundancy in the routing table? What's the premise of the existence of redundancy? After data mining of the routing tables, we find that although the routing table is rapidly growing (*some* backbone routers have more than 400K FIB entries today), the port number of a router is very limited (ranging from 3 to 80) and almost static. This observation intuitionally gives a positive answer to the existence of redundancy. Fortunately, the redundancy caused by the almighty gap between the prefix number and port number in the routing table can be quantized by ***Pigeonhole Principle***. Based on this observation, we also deduce the bound of the worst case of the FIB compression solution in this paper.

b) Correctness. After a profound study, we find that the LPM rule can be well expressed by the regular expression syntax. We also find that the LPM rule can be well expressed by the ***Group*** theory. Based on these two advancements, two basic equivalent atomic models are induced -- *election* model and *representative* model. We insist that all the trie-transformation algorithms can be proven by these two fundamental atomic models.

Actually, FIB compression algorithm is a tough task and is error-prone during the algorithm design and implementation. In order to guarantee the *ultimate correctness* of FIB compression algorithms, we propose Routing Table Equation Test (RTET) to verify the equivalence of the two routing tables before and after compression by traversing the 32-bit IP address space.

Specifically, the main contributions of this paper lie in the following aspects:

- We propose a universal mathematical method based on a new defined ***Group***, and apply this method to four classical FIB compression algorithms.
- We compute the bound of the worst case of FIB compression solution.
- We propose and implement Routing Table Equation Test (RTET) for the first time, to verify the

[1] This work is supported by NSFC (61202489), and the National Science & Technology Pillar Program No.2012BAH01B03, and the Instrument Developing Project of CAS under Grant No.YZ201229.

[2]Group (mathematics) [20] is a set together with a binary operation satisfying certain algebraic conditions.

[3]Both FIB compression and overlap elimination algorithms transform the binary trie, thus they are called trie-transformation algorithms in this paper.

978-1-4799-0913-1/14/$31.00 ©2014 IEEE

equivalence of the two tries before and after binary trie transformation by traversing the 32-bit IP address space.

II. Mathematic Proof

A. Group Definition

Prefixes are a series of bits. It can be well represented by regular expression syntax [19], and the symbols frequently used in this paper are defined below:

- A is a node in the trie, while (A) represents node A's prefix. Solid nodes have next-hop, while hollow nodes haven't.
- (AB) represents the bit string of the path between node A and B, while no solid nodes appear in the path.
- If A is an ancestor of B, then A ⊂B
- L(A) represents the prefix length of node A.
- P represents a trie, and (A) represents a prefix, then P(A) means the next-hop of prefix (A) in trie P.

Definition 1. *LPM **Group**.*

Let G be the LPM ***Group,*** and G=Z. The operation on LPM ***Group*** is XOR:

$$\forall x, y \in G$$

$$\boldsymbol{x\oplus y}=\begin{cases}\boldsymbol{x+y}, & \boldsymbol{xy(x+y)=0}\\ \boldsymbol{y}, & \boldsymbol{x>0, y>0}\\ \boldsymbol{meaningless}, & \boldsymbol{other\ else}\end{cases}$$

As shown in Figure 1, the function $\boldsymbol{z=x\oplus y}$ is plotted in three-dimensional space.

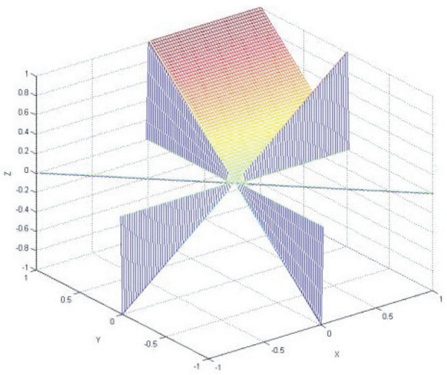


Figure 1. LPM **Group** in three-dimensional space.

Condition 1. *Closure*

Proof.

$$\forall x, y \in G, \text{obviously}, x\oplus y \in G.$$

Therefore, LPM **Group** satisfies *Closure*. □

Condition 2. *Associativity*

Proof.

∀x, y, z ∈ G

$$(x\oplus y)\oplus z = x\oplus(y\oplus z).$$

1) If $x=0$, $(x\oplus y)\oplus z = y\oplus z, x\oplus(y\oplus z) = y\oplus z$.

Therefore,

$$(x\oplus y)\oplus z = x\oplus(y\oplus z).$$

Similarly, if $y=0$ or $z=0$, $(x\oplus y)\oplus z = x\oplus(y\oplus z)$.

2) x ≠ 0 and y ≠ 0and z ≠ 0.

2.1) If $x+y=0$, in order to make $(x\oplus y)\oplus z$ and $x\oplus(y\oplus z)$ meaningful, $y+z$ must be zero. Therefore,

$$(x\oplus y)\oplus z = 0\oplus z = z,$$
$$x\oplus(y\oplus z) = x\oplus 0 = x = z.$$
$$\therefore (x\oplus y)\oplus z = x\oplus(y\oplus z).$$

2.2) If $x>0, and\ y>0$

$$(x\oplus y)\oplus z = y\oplus z = z,$$
$$x\oplus(y\oplus z) = x\oplus z = z.$$
$$\therefore (x\oplus y)\oplus z = x\oplus(y\oplus z).$$

Therefore, LPM **Group** satisfies *Associativity*. □

Condition 3. *Identity*

Proof.

$$\left.\begin{aligned} 0\oplus y &= \begin{cases}0, & y=0\\ y, & y>0\end{cases} \Rightarrow 0\oplus y = y\\ y\oplus 0 &= \begin{cases}0, & y=0\\ y, & y>0\end{cases} \Rightarrow y\oplus 0 = y\end{aligned}\right\}$$

$$\Rightarrow 0 \text{ is the identity.}$$

Therefore, LPM **Group** satisfies *Identity*. □

Condition 4. *Invertibility*

Proof.

$$\forall x \in G, x\oplus(-x) = (-x)\oplus x = 0.$$

Therefore, -x is the inverse of x. □

According to the above four conditions, it can be concluded that G is a **Group**. □

*LPM **Group*** is used to describe the matching process and results of prefixes in this paper, and thus we define the next-hop and induce Theorem 1 in the following.

Definition 2. *P(R)*

∀IP address R, R=[0,1]{32}, the match result of each bit is $\mathrm{S_i}$, for IPv4, $\mathrm{i}=1,2,\ldots,32$; for IPv6, $\mathrm{i}=1,2,\ldots,128$; According to the Longest Prefix Matching rule, the next-hop of R is $P(R) = S_1\oplus S_2\oplus S_3\oplus \ldots S_{32} = \oplus_{i=1}^{32} S_i$.

Theorem 1. *If the match results of every section of two prefixes are same, then the next-hops of the two prefixes are same.*

Proof.

$$P1(R) = \oplus_{i=1}^{32} S_i$$
$$= (\oplus_{i=1}^{t1} S_i)\oplus(\oplus_{i=t1+1}^{t2} S_i)\oplus(\oplus_{i=t2+1}^{t3} S_i)\oplus \ldots \oplus(\oplus_{i=tn+1}^{32} S_i)$$

$$P2(R) = \oplus_{i=1}^{32} V_i$$
$$= (\oplus_{i=1}^{t1} V_i)\oplus(\oplus_{i=t1+1}^{t2} V_i)\oplus(\oplus_{i=t2+1}^{t3} V_i)\oplus \ldots \oplus(\oplus_{i=tn+1}^{32} V_i)$$

Suppose $P1_k = \oplus_{i=tx+1}^{k} S_i, P2_k = \oplus_{i=tx+1}^{k} V_i$, then

$$P1(R) = P1_{t1}\oplus P1_{t2}\oplus \ldots \oplus P1_{32}$$

$$P2(R) = P2_{t1} \oplus P2_{t2} \oplus \dots \oplus P2_{32}$$

$$P1_k = P2_k, k = t1, t2, \dots, 32$$

Therefore, $P1(R) = P2(R)$. □

This theorem can be used to prove the equivalence of the next-hop of two tries section by section with regard to one IP address.

Theorem 2. *Decision Theorem*

The necessary and sufficient condition that two tries are equivalent is the next-hops are equal in the two trie for any IP addresses by LPM rule.

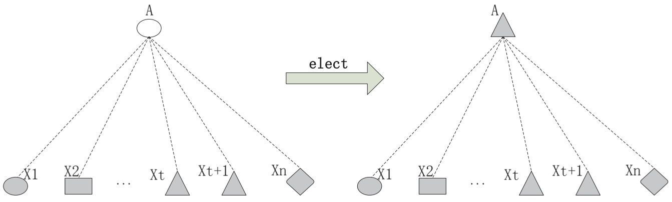
Obviously, this *Decision Theorem* naturally holds. Combining Theorem 1 and Theorem 2, we can prove the equivalence of two trie (or two models) section by section.

B. Election and Representative Models

We insist that all the trie-transformation algorithms can be proven by two basic transformation models: election model and representative model.

1) Election Model

Election Model: two or more nodes elect their common ancestor node, and no solid node appears in the path from the candidate nodes to the common ancestor node. Any candidates can be elected as representative, resulting in different compression ratio.



(a) The original trie (b) Trie after election

Figure 2. Election Model.

Election models can work on both binary trie and multi-bit trie. As shown in Figure 2, the next-hop of Node Xi is Ni, the count of Ni is Ci.

Election result: if such t exists: $\forall$ j! = t, Cj $\leq$ Ct holds, then Xt is the elected representative. If such t doesn't exist, election fails. Then the common ancestor's next-hop is set to NULL, and participates in the next round election. In this way, an optimal compression ratio can be achieved.

Proof.

$\forall$ IP address R, obviously, $L(R) = K, R = [0,n]\{K\}$. Suppose $R = [0,n]\{L(A)\}[0,n][0,n]\{K - L(A) - 1\}$.

Step1: match $[0,1]\{L(A)\}$

$$\left.\begin{array}{r} [0,1]\{L(A)\} = (A) \Rightarrow \begin{cases} P1([0,1]\{L(A)\}) = P1(\tilde{A}) \\ P2([0,1]\{L(A)\}) = P2(A) \end{cases} \\ [0,1]\{L(A)\} \neq (A) \Rightarrow \begin{cases} P1([0,1]\{L(A)\}) = P1(\tilde{A}) \\ P2([0,1]\{L(A)\}) = P2(\tilde{A}) \end{cases} \\ P1(A) = P2(A) \\ P1(\tilde{A}) = P2(\tilde{A}) \end{array}\right\}$$

$$\Rightarrow \begin{cases} P1([0,1]\{L(A)\}) = P1_{s1} \\ P2([0,1]\{L(A)\}) = P2_{s1} \end{cases}$$

Step2: match [0,1]

$$\left.\begin{array}{r} [0,1] = i, i = 1,2,3, \dots, n \Rightarrow \begin{cases} P1([0,1]) = P1(X_i) \\ P2([0,1]) = P2(X_i) \end{cases} \\ P1(X_i) = P2(X_i) \end{array}\right\}$$

$$\Rightarrow P1([0,1]) = P2([0,1]) = P_{s2} \neq 0$$

Step3: match $[0,1]\{K - L(A) - 1\}$

$$\left.\begin{array}{r} [0,1] = i, i = 1,2, \dots, n \\ \Rightarrow \begin{cases} P1([0,n]\{K - L(A) - 1\}) = P1(X_i *) \\ P2([0,n]\{K - L(A) - 1\}) = P2(X_i *) \end{cases} \\ P1(X_i *) = P2(X_i *) \end{array}\right\}$$

$$\Rightarrow P1([0,n]\{K - L(A) - 1\}) = P2([0,n]\{K - L(A) - 1\}) = P_{s3}$$

According to step1, step2, and step3,

$$\begin{aligned} P1(R) &= P1([0,n]\{L(A)\}[0,n][0,n]\{K - L(A) - 1\}) \\ &= P1([0,n]\{L(A)\}) \oplus P1([0,n]) \oplus P1([0,n]\{K - L(A) - 1\}) \\ &= P1_{s1} \oplus P_{s2} \oplus P_{s3} \end{aligned}$$

$$\begin{aligned} P2(R) &= P2([0,n]\{L(A)\}[0,n][0,n]\{K - L(A) - 1\}) \\ &= P2([0,n]\{L(A)\}) \oplus P2([0,n]) \oplus P2([0,n]\{K - L(A) - 1\}) \\ &= P2_{s1} \oplus P_{s2} \oplus P_{s3} \end{aligned}$$

$\because P_{s2} \neq 0$, according to the associative law,

$$P1(R) = P1_{s1} \oplus P_{s2} \oplus P_{s3} = (P1_{s1} \oplus P_{s2}) \oplus P_{s3} = P_{s2} \oplus P_{s3}$$

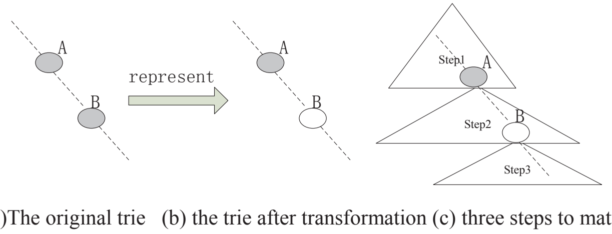
$$P2(R) = P2_{s1} \oplus P_{s2} \oplus P_{s3} = (P2_{s1} \oplus P_{s2}) \oplus P_{s3} = P_{s2} \oplus P_{s3}$$

$$\therefore P1(R) = P2(R)$$

According to Theorem 1 and Theorem 2, $P1 \Leftrightarrow P2$. □

If P2 is the election model of P1, we say P2 = Ele(P1). Actually, any node can be elected as representative, resulting in different compression ratio, and the proof method is similar.

2) Representative Model



(a)The original trie (b) the trie after transformation (c) three steps to match

Figure 3. Representative model.

Representative: after a successful election, the common ancestor will exercise the right of representative immediately: set the next-hop of its voters (those candidates which own the same next-hop with representative) to 0. As shown in Figure 3, the next-hop of A and B is same, and A is the nearest ancestor of B. In this case, B's next-hop is set to zero. The proof is similar to that of election model, thus is ignored.

If P2 is the representative model of P1, we say $\mathrm{P2} = \mathrm{Rep(P1)}$. We insist that all models can be proved by election model and representative model.

III. THE WORST CASE OF FIB COMPRESSION SOLUTION

In this section, the bound of the worst case of FIB compression solution is computed, so as to prove the feasibility and effectiveness of FIB compression algorithms.

A) Pigeonhole Principle

In mathematics, the ***Pigeonhole Principle*** states that if n+1 objects are distributed into n boxes, then at least one box contains two or more of the objects [21]. This is a simple but very useful principle. For example, if there are five people from four countries, there are at least two people from the same country.

B) *The Worst Case for Full IP Address Space*

For IPV4，the space is 2^{32}. Suppose there are 30 ports and 2^{32} prefixes with the length of 32 (full IP address space) in a routing table. At level 32, every 32 nodes elect their common ancestor. At least two ports are the same according to the ***Pigeonhole Principle***. Therefore, at least two nodes of 32 nodes can be compressed into one, and thus $2^{32}/32 = 2^{27}$ nodes are reduced. At level 27 of the trie, there are 2^{27} nodes. Similarly, 32 nodes select their common ancestor. According to the ***Pigeonhole Principle***, at least two nodes can be compressed into one, and $2^{27}/32 = 2^{22}$ nodes are reduced. Therefore, the number of left nodes is at least

$$R = 2^{32} - \frac{2^{32}}{2^{5\times1}} - \frac{2^{32}}{2^{5\times2}} - \frac{2^{32}}{2^{5\times3}} - \cdots - \frac{2^{32}}{2^{5\times6}} \qquad (1)$$

This worst case exists – if the preorder traverse results are Ni (i=1, 2, 3…), and the next-hop of Ni is represented by P(Ni), which satisfies:

$$\mathrm{P(Ni)} = \mathrm{i} \bmod (32)$$

In this case, the number of compressed routing table by optimal algorithm is R in equation (1).

IV. ROUTING TABLE EQUATION TEST

The mathematical proof method has been elaborated above, but there might be flaws in the process of mathematical derivation and coding. How to guarantee the *ultimate correctness* of these algorithms? The *ultimate correctness* refers to that for any IP address, the compressed routing table tells the same next-hop with the original table. Therefore, we propose Routing Table Equation Test (RTET) to judge the equivalence of the two routing tables. RTET firstly builds two tries, then traverses 32-bit IP address space, and compares the next-hop of two tries by using the same IP address. If and only if all are equal, the two routing tables are equivalent. Otherwise, RTET stops and tells the prefix and the different next-hop of the two tries. One comparison of two routing tables by using RTET takes about 16 minutes. The algorithms [2-5] are all implemented and verified by RTET, using the routing tables downloaded from [22].

V. CONCLUSIONS

FIB compression has been a hot topic of scientific research for years. There are many FIB compression and overlap elimination algorithms, but there isn't a formal and universal mathematical method to guarantee their correctness. Therefore, we propose a universal mathematical method for trie-transformation algorithms based on a new defined ***Group***.

REFERENCES

[1] X. Meng, Z. Xu, B. Zhang, G. Huston, S. Lu, and L. Zhang, IPv4 Address Allocation and the BGP Routing Table. ACM SIGCOMM Computer Communication Review, vol. 35, pp. 71–80, 2005.

[2] R.Draves, C.King, S.Venkatachary, and B.D.Zill. Constructing Optimal IP Routing Tables. In Proc. IEEE INFOCOM, pp. 88–97, 1999.

[3] B. Cain. Auto aggregation method for IP prefix/length pairs. http://www.patentgenius.com/patent/6401130.html. 2002.

[4] X. Zhao, Y. Liu, L. Wang, and B. Zhang. On the Aggregatability of Router Forwarding Tables. In Proc. IEEE INFOCOM, 2010.

[5] Yaoqing Liu, Xin Zhao, Kyuhan Nam, Lan Wang, Beichuan Zhang. Incremental Forwarding Table Aggregation. In Proc. IEEEE GLOBECOM, 2010.

[6] Qing Li, Dan Wangy, Mingwei Xu, Jiahai Yang. On the Scalability of Router Forwarding Tables: Nexthop-Selectable FIB Aggregation. In Proc. IEEE INFOCOM, 2011.

[7] Heeyeol Yu. A memory- and time-efficient on-chip TCAM minimizer for IP lookup.DATE '10 Proceedings of the Conference on Design, Automation and Test in Europe2010.

[8] M. Waldvogel, G. Varghese, J. Turner and B. Plattner. Scalable High Speed IP Routing Lookups. Computer Communications Review, October 1997.

[9] Degermaerk, M., Brodnik, A., Carlsson, S., and Pink, S. Small forwarding tables for fast routing lookups. In Proc. SIGCOMM, NY, 1997

[10] S. Nilsson and G. Karlsson. Fast Address Look-up for Internet Routers Proceedings of IEEE Broadband Communications, April 1998.

[11] Zheng, K., Hu, C., Lu, H., Liu, B. A TCAM-based distributed parallel IP lookup scheme and performance analysis. IEEE/ACM Trans. Netw. 14, 863–875, 2006.

[12] Lin, D., Zhang, Y., Hu, C., Liu, B., Zhang, X., Pao, D. Route Table Partitioning and Load Balancing for Parallel Searching with TCAMs. In Proc. IPDPS, 2007.

[13] Tong Yang, Ruian Duan, Jianyuan Lu, Shenjiang Zhang, Huichen Dai and Bin Liu. CLUE: achieving fast update over compressed table for parallel lookup with reduced dynamic redundancy. Accepted by Proc. IEEE ICDCS, 2012.

[14] V. Srinivasan and G. Varghese, Fast IP lookups using controlled prefix expansion, ACM TOCS, vol. 17, pp. 1–40, 1999.

[15] Tong Yang, Ting Zhang, Shenjiang Zhang and Bin Liu. Constructing Optimal Non-overlap Routing Tables. Accepted by Proc. IEEE ICC, 2012.

[16] Miguel Á. Ruiz-Sánchez, Ernst W. Biersack, Walid Dabbous. Survey and Taxonomy of IP Address Lookup Algorithms. Network, IEEE. 2001

[17] IRTF Routing Research Group. http://www.irtf.org/charter?gtype=rg/&group=rrg.

[18] IETF Global Routing Operations (GROW). http://datatracker.ietf.org/wg/grow/charter/.

[19] http://www.regular-expressions.info/reference.html

[20] D. D. Vvedensky. Group Theory. World Scientific Pub. pp. 14-15. 2005

[21] Brualdi, R. A. Introductory Combinatorics, Fifth Edition. China Machine Press. Chapter 3, pp. 69-70, 2009

[22] RIPE Network Coordination Centre. http://www.ripe.net/data-tools/stats/ris/ris-raw-data.