Towards TCAM-based Scalable Virtual Routers

Layong Luo, Gaogang Xie (ICT, CAS)

Steve Uhlig (QMUL)

Laurent Mathy (U. of Liège/Lancaster U)

Kavé Salamatian (U. of Savoie)

Yingke Xie (ICT, CAS)

Motivation

- Virtual routers (VRs)
  - key building blocks for enabling network virtualization
  - VPN, network testbeds …
- Memory scalability issue
  - The number of FIBs, and the size of each FIB, are expected to increase continuously
  - FIBs are preferably stored in high-speed memory (SRAMs or TCAMs) with limited size

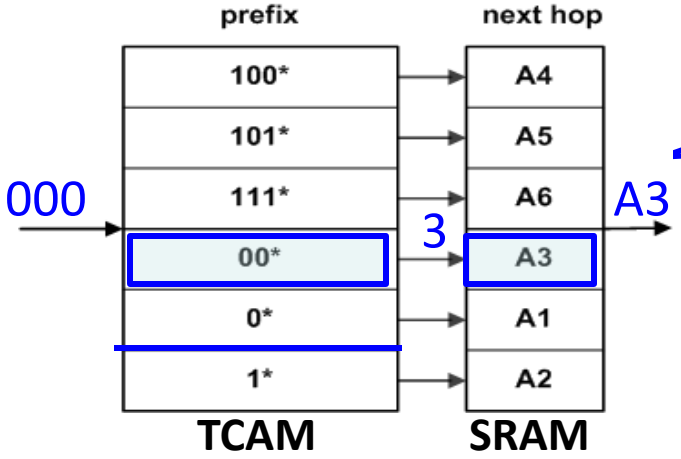
How to support as many FIBs as possible in the limited high-speed memory?

Related work

- SRAM-based scalable virtual routers
  - Trie overlap, CoNEXT 2008
  - Trie braiding, INFOCOM 2010
  - Multiroot, ICC 2011
  - …
- None of previous work has exploited the possibility of using TCAMs to build scalable virtual routers

Background

- Traditional TCAM-based IP lookup engine



one lookup per clock cycle in any case!

- Non-shared approach for TCAM-based virtual routers

FIB 0 VID 0

prefix	next hop
0*	A1
1*	A2
00*	A3
100*	A4
101*	A5
111*	A6

(a)

FIB 1 VID 1

prefix	next hop
0*	B1
1*	B2
11*	B3
100*	B4
101*	B5
111*	B6

(b)

prefix	next hop
0100*	A4
0101*	A5
0111*	A6
000*	A3
00*	A1
01*	A2
1100*	B4
1101*	B5
1111*	B6
111*	B3
10*	B1
11*	B2

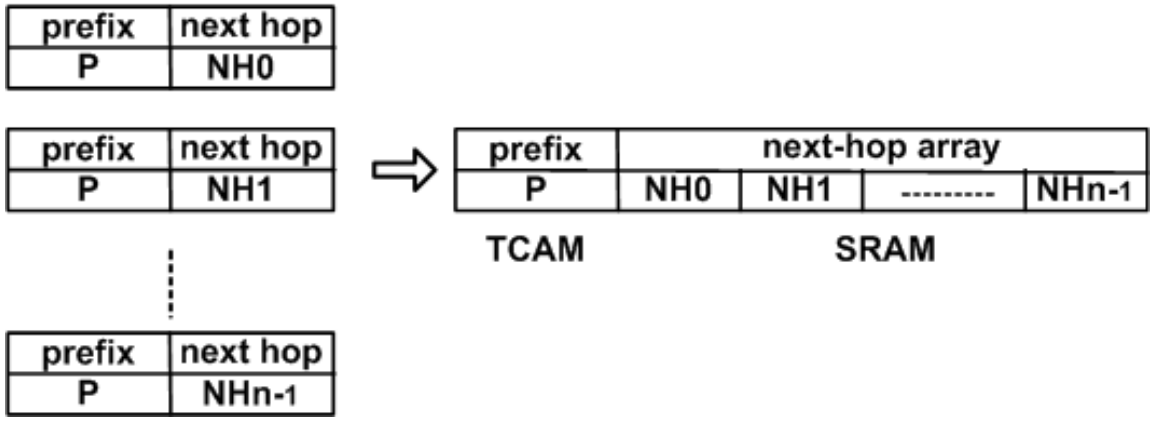
TCAM SRAM

Poor scalability:

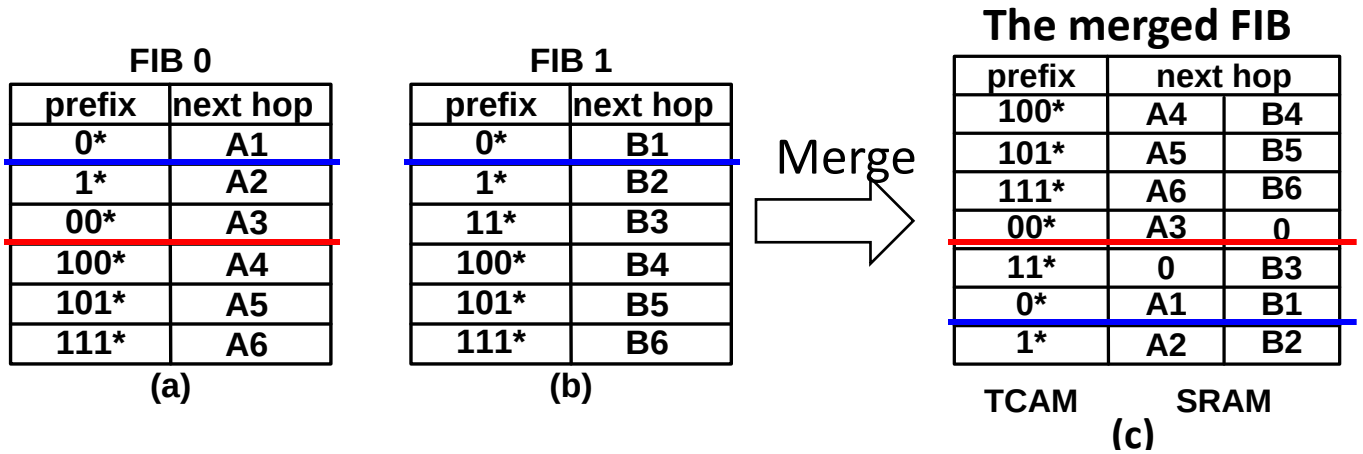
$$S = \sum_{i=1}^{n} S_i$$

Institute of Computing Technology, Chinese Academy of Sciences

Merged data structure



An example:

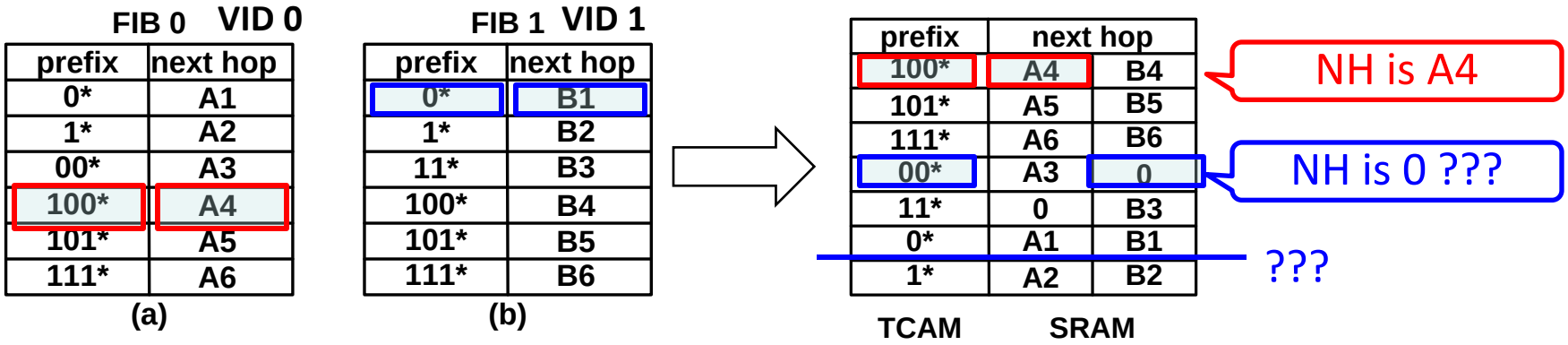


Total 12 TCAM entries **vs.** **Only 7 TCAM entries**

Lookup issue

- Example 1: IP 100, VID 0 (correct lookup)

  Example 2: IP 000, VID 1 (incorrect lookup)



- Prefix masking issue
  - *Incorrect matching, resulting from the masking of a shorter prefix (e.g., <0*, B1>) in an original FIB, by a longer prefix (e.g., <00*, 0>) in the merged FIB, must be avoided*

Solutions

Two TCAM FIB merging approaches

- FIB Completion
- FIB Splitting

FIB completion

- Basic idea
  - *Whenever a prefix from the merged FIB doesn't appear in a given individual FIB, we simply associate it with a valid NH in this FIB according to the LPM rule*

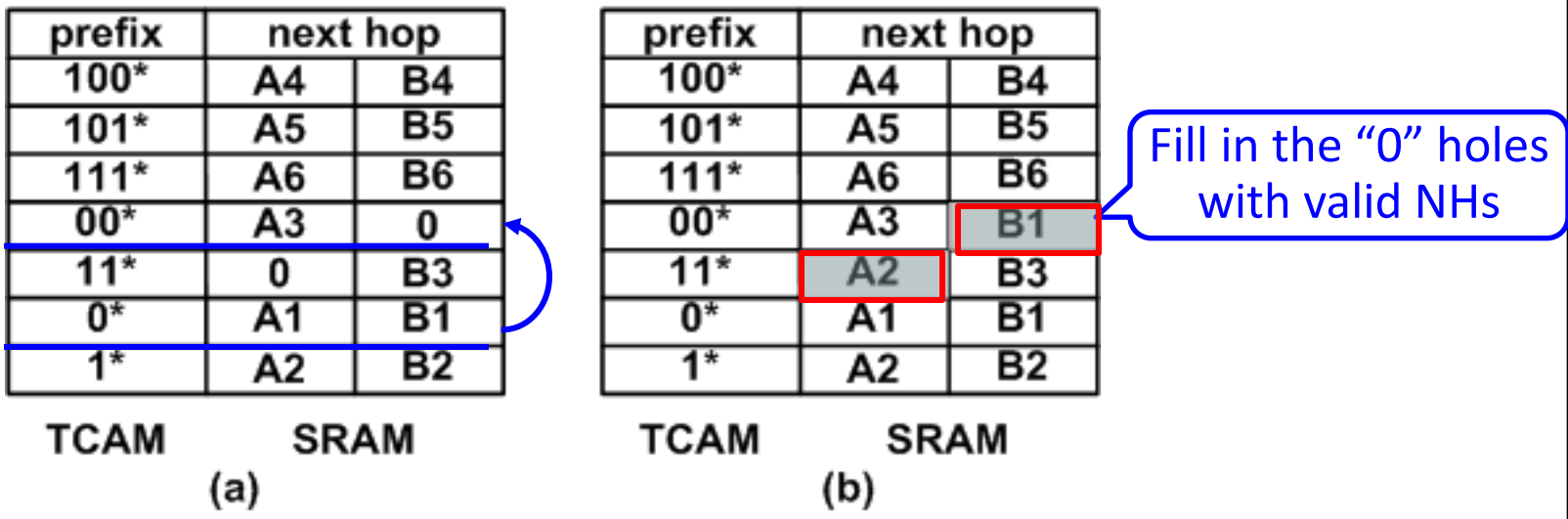


Fig. 1. (a) The basic merged FIB, and (b) its completed version

Completion process

- Auxiliary tries in software help the completion process

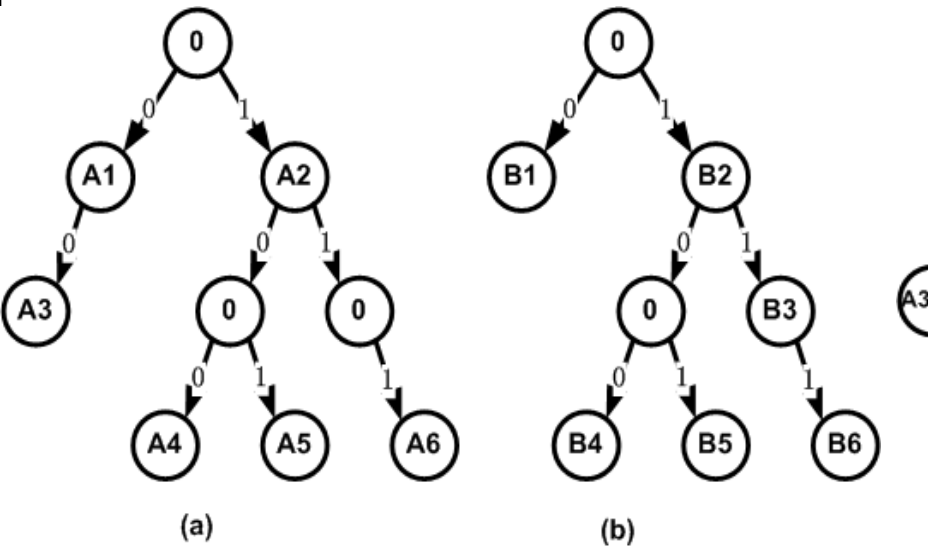


Fig. 1. two tries built from the two sample FIBs

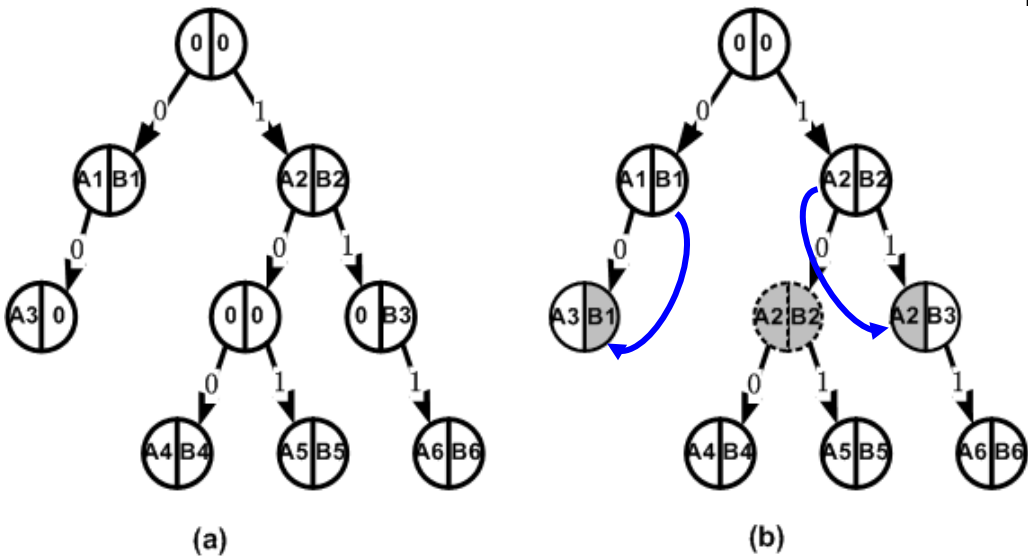
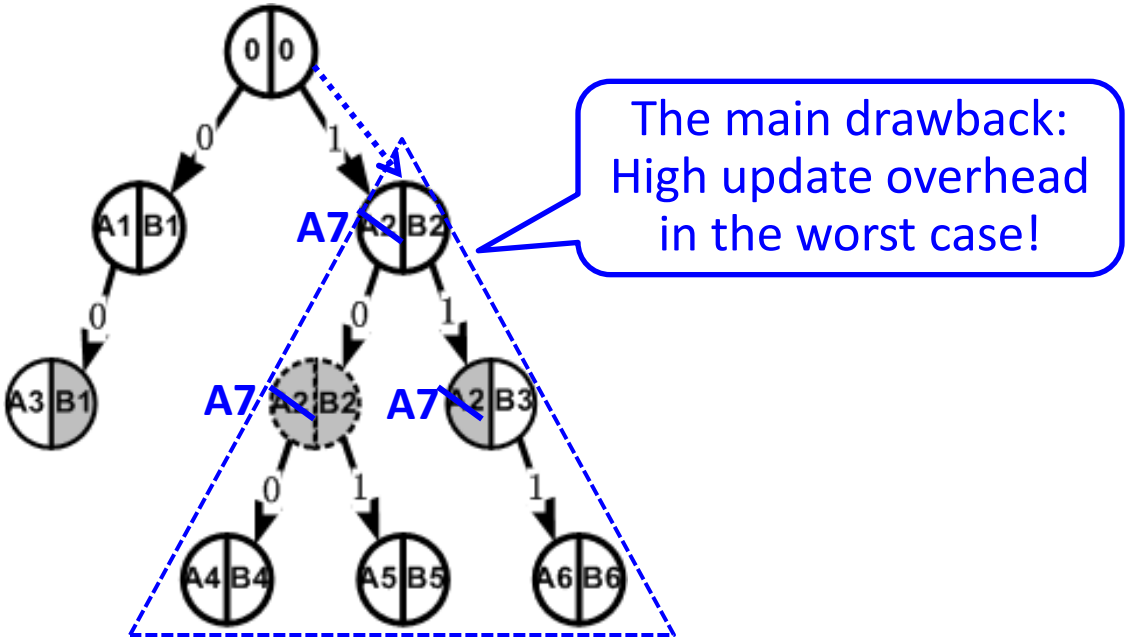


Fig. 2. (a) a merged trie using trie overlap[1] , and (b) its completed version

[1] J. Fu and J. Rexford, Efficient IP-address lookup with a shared forwarding table for multiple virtual routers, CoNEXT 2008

Update process

- Three steps
  - Update the auxiliary merged trie [in software]
  - Perform masking prefix correction [in software]
  - Modify the prefixes and NHs in the lookup engine [in hardware]
- An example: modify <1*, A2> with <1*, A7>



(a) An original merged trie in FIB completion

FIB splitting

- Naturally disjoint leaf prefixes, which are about 90% of the total prefixes, are merged in one TCAM
- The remaining small overlapping prefix set is stored in another TCAM using the non-shared approach

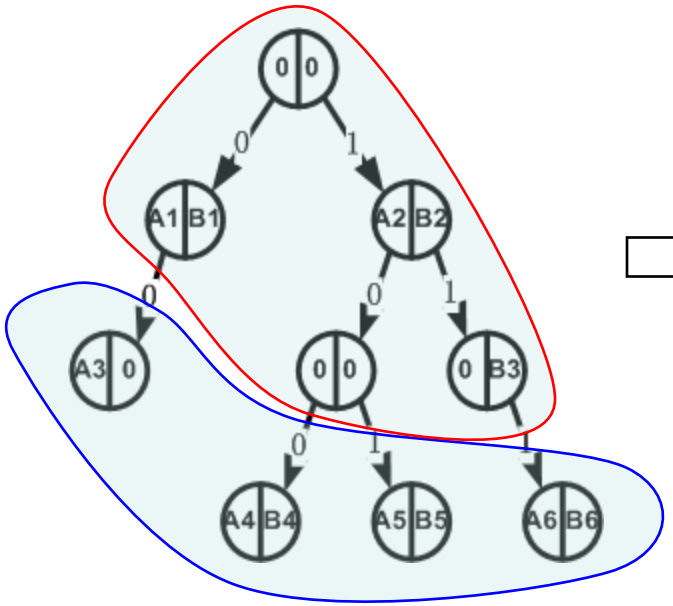


Fig. 1. A merged trie

prefix	next hop	
00*	A3	0
100*	A4	B4
101*	A5	B5
111*	A6	B6

(a)

prefix	next hop
00*	A1
01*	A2
111*	B3
10*	B1
11*	B2

(b)

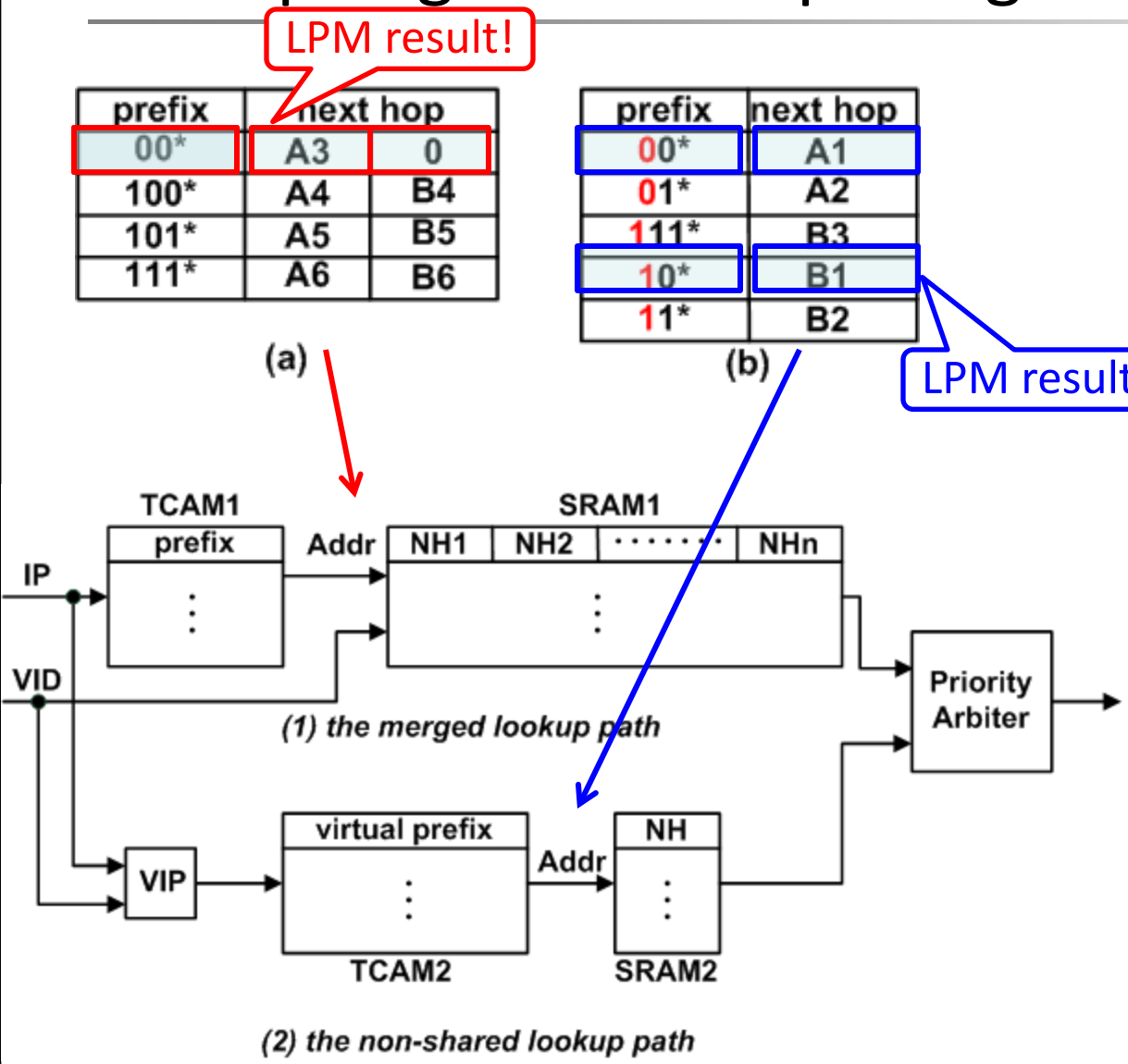
Prefix masking issue cannot exist in a disjoint prefix set

…int prefix set, and
…pping prefix set

Layong Luo, Gaogang Xie, Yingke Xie, Laurent Mathy and Kave Salamatian, A hybrid IP lookup architecture with fast updates, INFOCOM 2012

…ology,
…es

Lookup engine in FIB splitting



For a lookup

(1) If a valid NH is got from path 1, it is the correct LPM result. (e.g., IP 000, VID 0)

(2) If a valid NH is not got from path 1, the LPM result must be found in path 2. (e.g., IP 000, VID 1)

Update process

- Three steps
  - Update the auxiliary merged trie [in software]
  - Find changes in both prefix sets [in software]
  - Modify the prefixes and NHs in both lookup paths [in hardware]

- When compared to FIB completion
  - Prefix masking correction is totally avoided in FIB splitting

A more reasonable worst-case update overhead in FIB splitting!

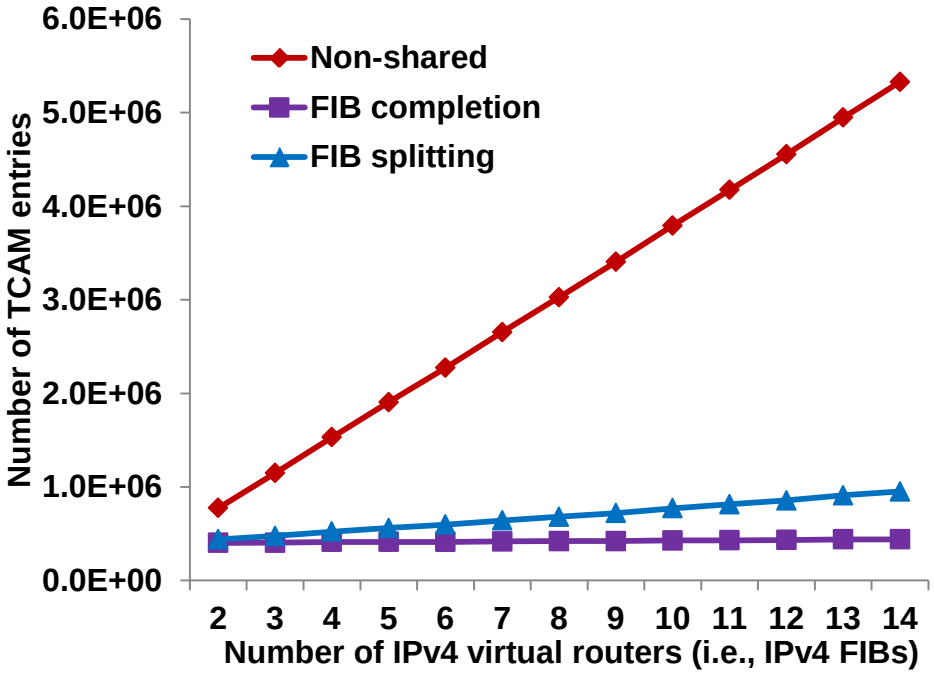
Performance evaluation

- Routing tables and update traces
  - 14 full routing tables from core routers [RIPE RIS Project]
  - 12 hours' update traces on these tables

- Comparison of the non-shared approach, FIB completion, and FIB splitting
  - TCAM size
  - SRAM size
  - Total cost of the system
  - Lookup and update performance

TCAM size

- Metric: the number of TCAM entries



For 14 IPv4 FIBs (each ~ 370 K – 400K entries):

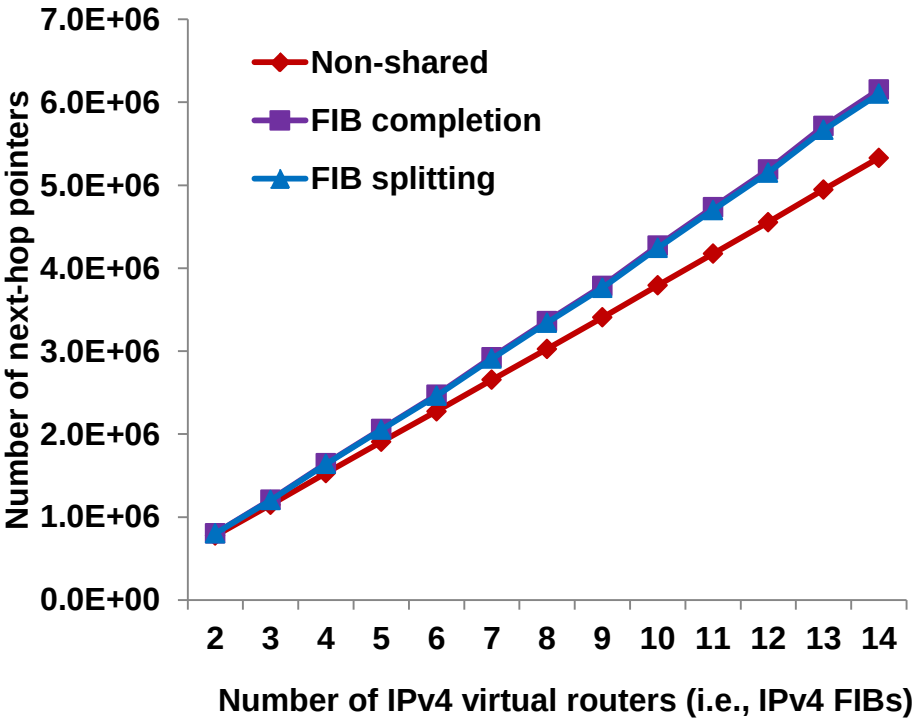
Non-shared: 5 M TCAM entries

FIB completion: 429 K TCAM entries (reduce by 92%)

FIB splitting: 928 K TCAM entries (reduce by 82%)

SRAM size

- Metric: the number of next hop pointers



For 14 IPv4 FIBs:

Non-shared:	40.7 Mb
FIB completion:	46.9 Mb (increase by 15%)
FIB splitting:	46.6 Mb (increase by 14%)

Total cost of the system

- A cost-effective tradeoff
  - Memory reduction of over 80% in expensive TCAMs
  - Memory Increase of roughly 15% in cheaper SRAMs

Table 2. Reference prices of TCAMs and SRAMs

Memory	Part No.	Capacity	Speed	Price
TCAM	NL9512	512K×40bit	250MHz	$387.2
SRAM	CY7C1525	8M×9bit	250MHz	$89.7

Table 3. Cost of the three approaches for IPv4 FIBs

	# of TCAMs	# of SRAMs	Total Cost
Non-shared	11	1	$4348.9
FIB completion	1	1	$476.9 → reduce by 89%
FIB splitting	3	2	$1341 → reduce by 69%

Lookup & update performance

- Metric
  - Lookup performance: the number of clock cycles per lookup
  - Update overhead: the number of write accesses per update

- Theoretical analysis

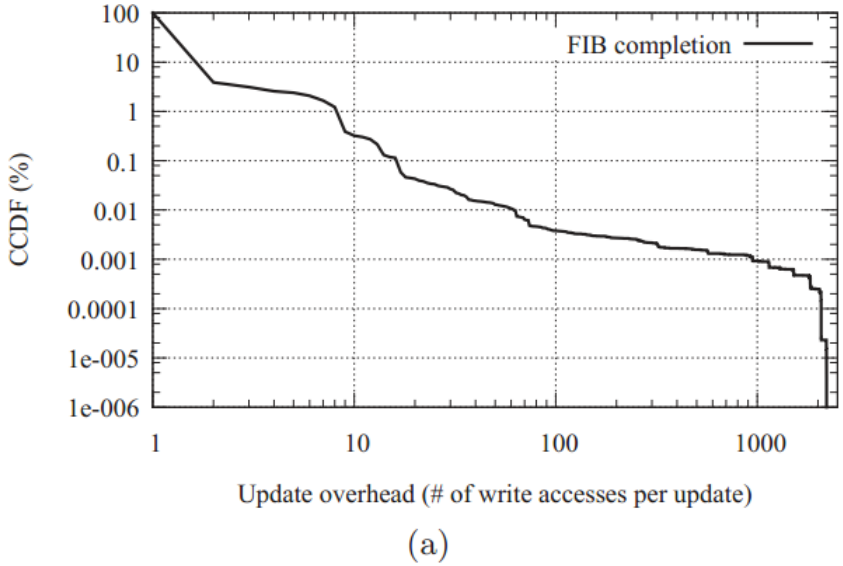
Table 5: Theoretical worst-case lookup performance and update overhead

	Lookup	Update
Non-shared	O(1)	W/2
FIB completion	O(1)	$2^{W+1} - 1$
FIB splitting	O(1)	NW/2

W: the length of the IP address
N: the number of virtual routers

Lookup & update performance

- Actual update overhead
  - 12 hours' update traces from RIPE RIS Project



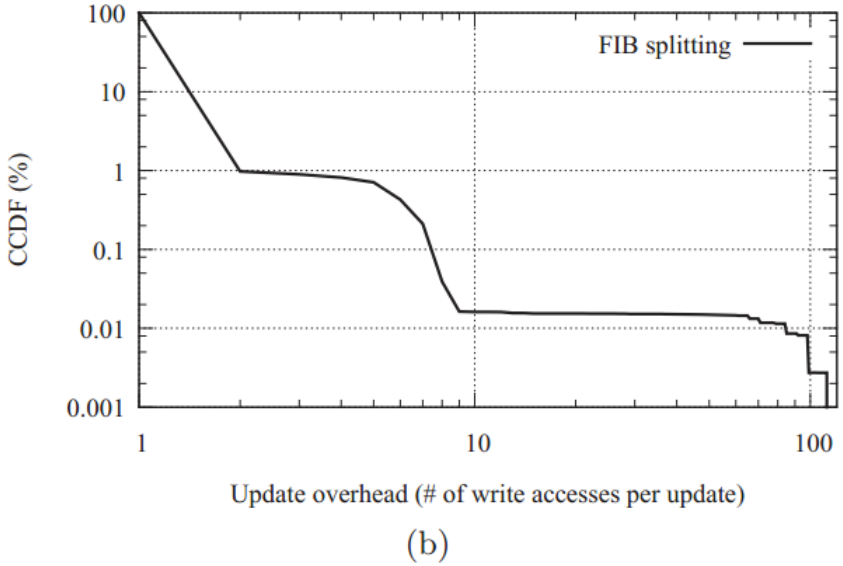


Figure 14: Complementary cumulative distribution function of update overhead in (a) FIB completion and (b) FIB splitting

Most updates cost only 1 write access per update, and large-overhead updates rarely happen in practice

Conclusions

- Main contributions
  - The first work to exploit the possibility of using TCAMs to build scalable virtual routers
  - Merged data structure and prefix masking issue
  - Two approaches with different tradeoffs
    - FIB completion: best scalability but high worst-case update overhead
    - FIB splitting: good scalability with a more reasonable upper bound on the worst-case update overhead
- Future work
  - Implementation on PEARL 2.0 platform [IEEE Commun. Mag. 2011]
  - Dissimilar FIBs

Thank you!

Acknowledgment

pFlower Project Granted by NSFC-ANR

SOFIA (Future Internet Architecture) Project Granted by MOST

More information: http://fi.ict.ac.cn

luolayong@ict.ac.cn