Towards TCAM-based scalable virtual routers

Layong Luo, Gaogang Xie, Yingke Xie (ICT/CAS)

Kavé Salamatian (U. of Savoie)

Steve Uhlig (QMUL)

Laurent Mathy (U. of Liège/Lancaster U)

Motivation

Virtual routers (VRs)

- key building blocks for enabling network virtualization
- VPN, network testbeds, Future Internet …

Memory 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) but limited amount of it
- Scalability challenge: support as many FIBs as possible in the limited high-speed memory?

Related work

- SRAM-based scalable virtual routers

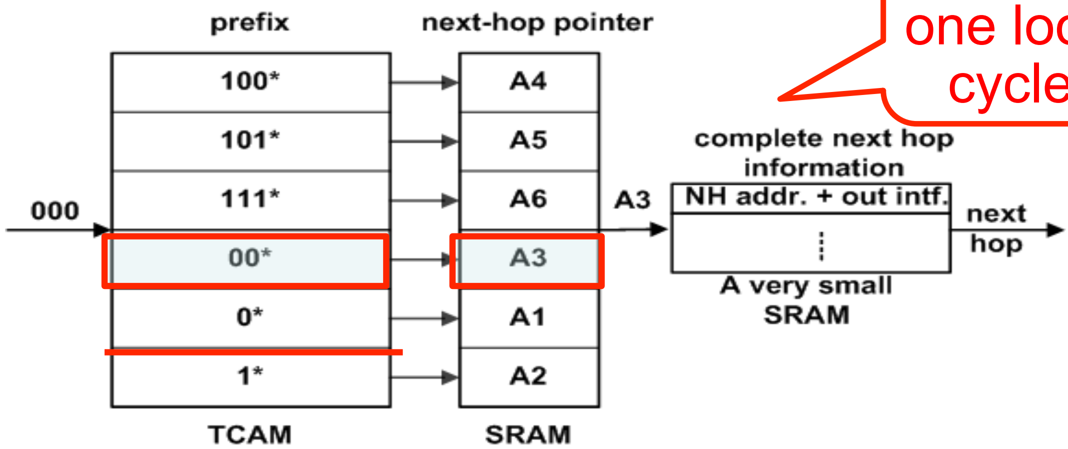
  (How to merge multiple tries to achieve good scalability?)
  - Trie overlap, CoNEXT 2008
  - Trie braiding, INFOCOM 2010
  - Multiroot, ICC 2011
  - ….

- None of previous work has exploited the possibility of using TCAMs to build scalable virtual routers

Background

- Traditional TCAM-based IP looku

The main advantage: 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:

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

Merged data structure

prefix	next hop
P	NH0

prefix	next hop
P	NH1

prefix	next hop
P	NHn-1

prefix	next-hop array			
P	NH0	NH1	---------	NHn-1
TCAM	SRAM			

An example:

FIB 0

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

(a)

FIB 1

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

(b)

The number of TCAM entries is significantly reduced!

prefix	next hop	
100*	A4	B4
101*	A5	B5
111*	A6	B6
00*	A3	0
11*	0	B3
0*	A1	B1
1*	A2	B2
TCAM	SRAM	

Lookup issue

- Example 1: IP 100, VID 0

  Example 2: IP 000, VID 1

FIB 0

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

(a)

FIB 1

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

(b)

prefix	next hop	
100*	A4	B4
101*	A5	B5
111*	A6	B6
00*	A3	0
11*	0	B3
0*	A1	B1
1*	A2	B2

TCAM SRAM

correct NH!

Incorrect NH!

???

- TCAM FIB merging principle
  - *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.*

Two approaches

Two FIB merging approaches that respect the principle

- 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.*

prefix	next hop	
100*	A4	B4
101*	A5	B5
111*	A6	B6
00*	A3	0
11*	0	B3
0*	A1	B1
1*	A2	B2

TCAM SRAM

(a)

prefix	next hop	
100*	A4	B4
101*	A5	B5
111*	A6	B6
00*	A3	B1
11*	A2	B3
0*	A1	B1
1*	A2	B2

TCAM SRAM

(b)

Fill in the "0" holes with valid NHs

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

Completion process

- Auxiliary tries help the completion process

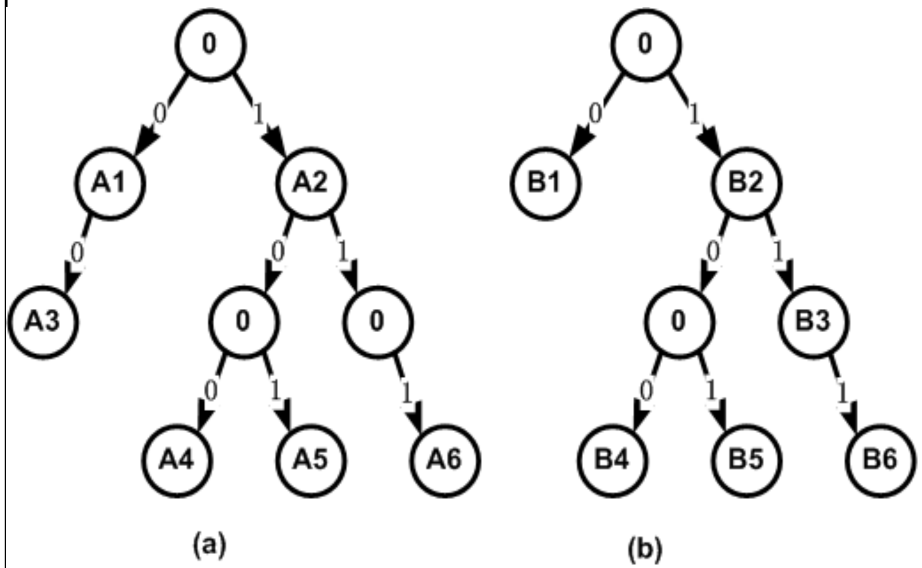


Fig. 1. two tries built from the two sample FIBs

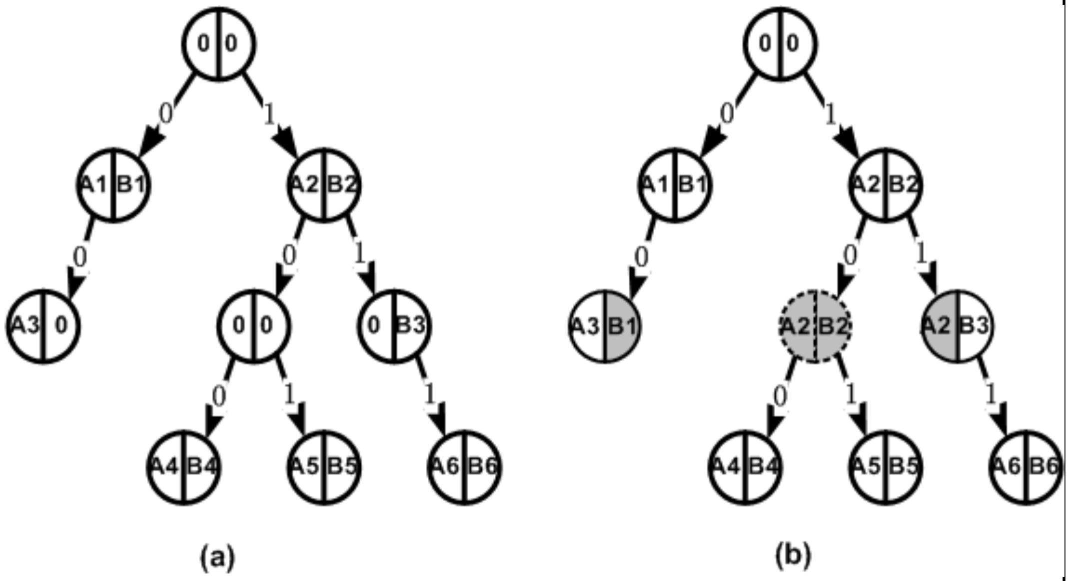
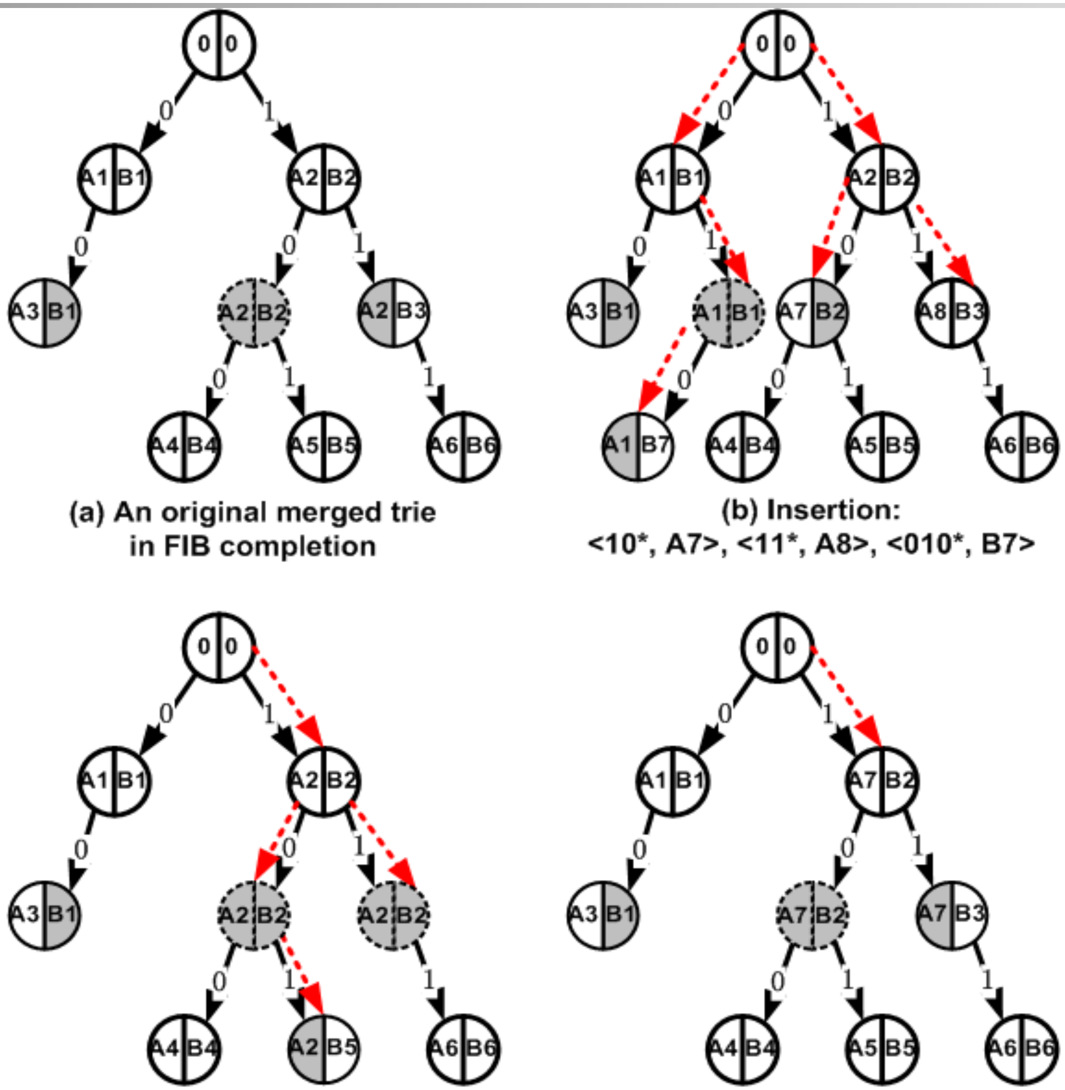


Fig. 2. (a) a merged trie using trie overlap[1] , and (b) its completed version

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

Update scenarios



(a) An original merged trie in FIB completion

(b) Insertion: <10*, A7>, <11*, A8>, <010*, B7>

(c) Deletion: <11*, B3>, <101*, A5>

(d) Modification: change <1*, A2> to <1*, A7>

FIB splitting

- The main drawback of FIB completion
  - High update overhead in the worst case due to the masking prefix correction process

- Another way to address the lookup issue is
  - To ensure that when a masking prefix is a hit for a lookup in the merged FIB, the corresponding masked prefix is also made available to the lookup process, rather than correct the NHs of masking prefixes, thus reducing the worst-case update overhead

Basic idea

- The 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

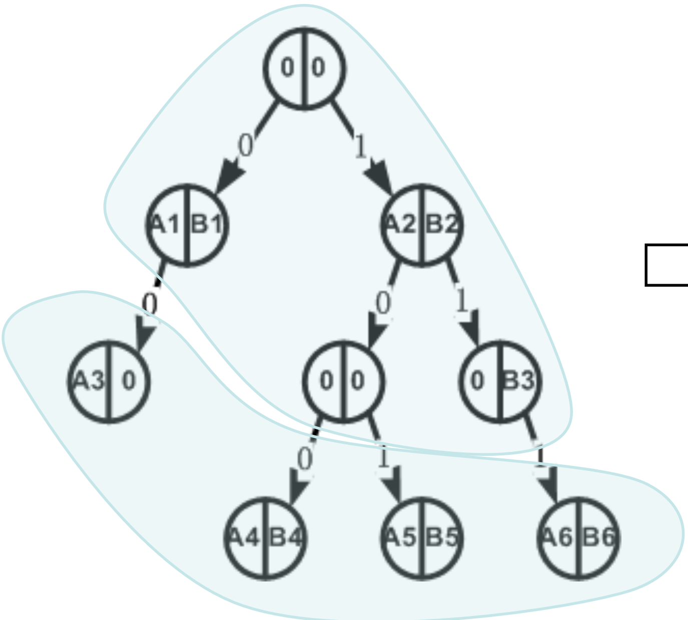


Fig. 1. A merged trie

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

(a)

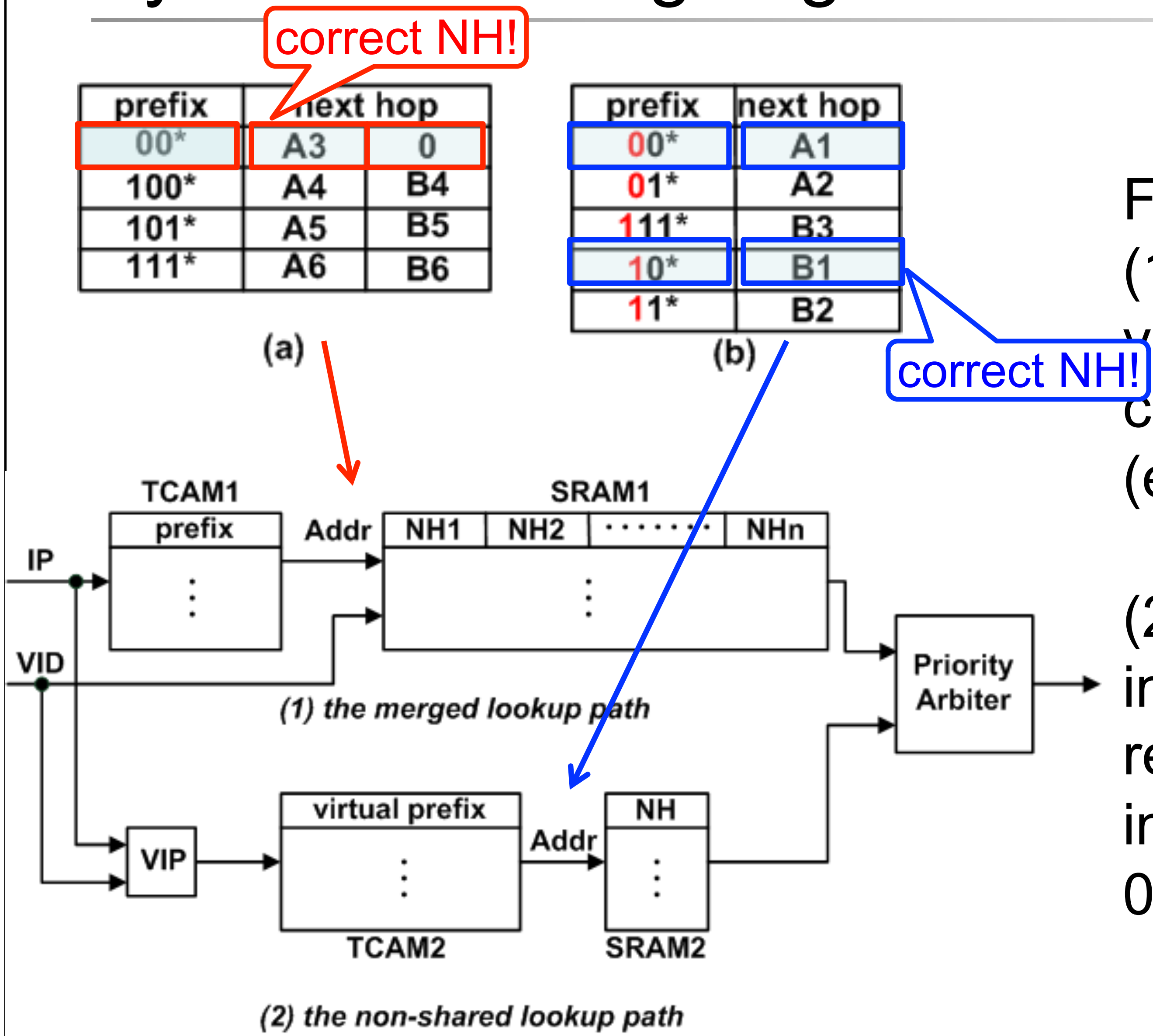
prefix	next hop
00*	A1
01*	A2
111*	B3
10*	B1
11*	B2

(b)

Masking problem cannot exist in a disjoint prefix set

oint prefix set,
verlapping
prefix set

Hybrid forwarding engine

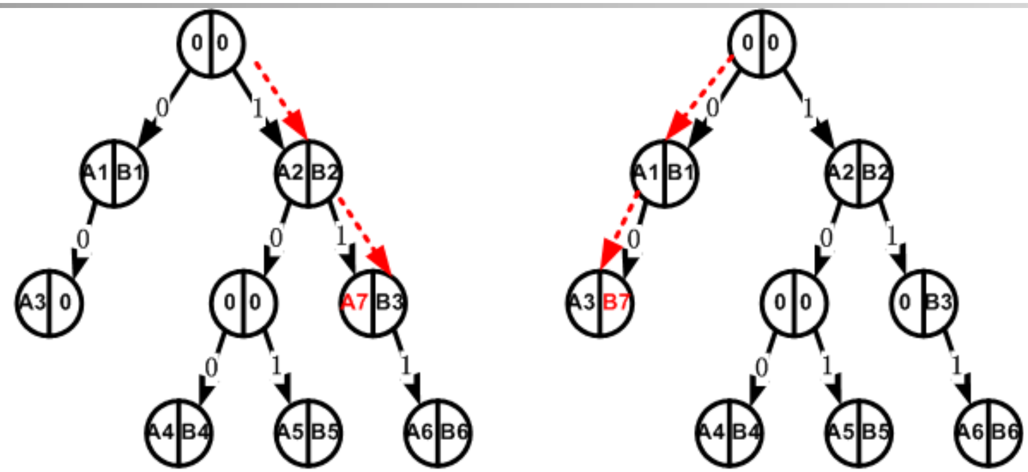


For a lookup

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

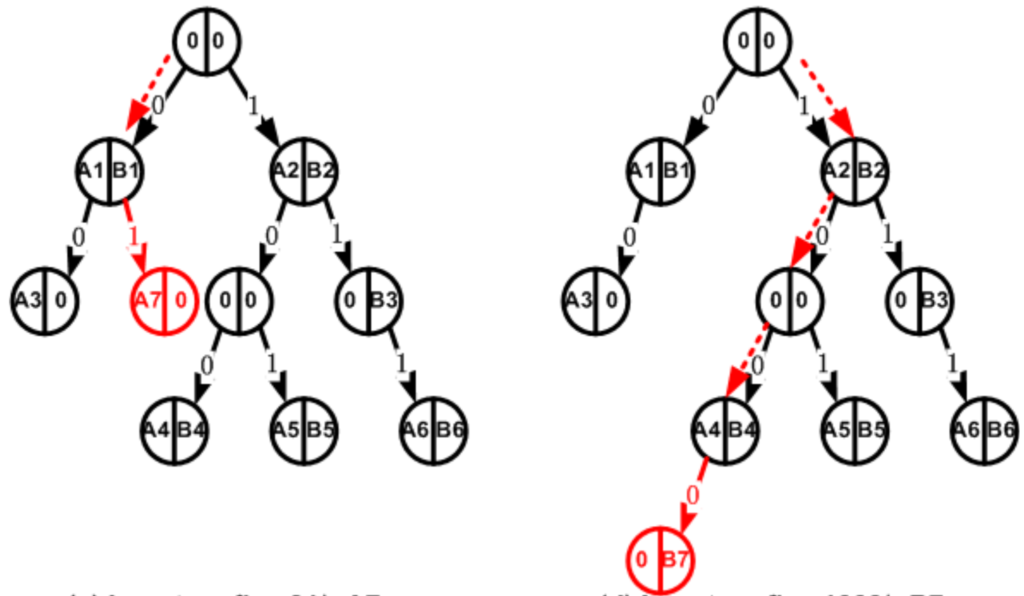
(2)If path 1 yields invalid NH, the LPM result must be found in path 2. (e.g., IP 000, VID 1)

Update scenarios



(a) Insert prefix <11*, A7>

(b) Insert prefix <00*, B7>



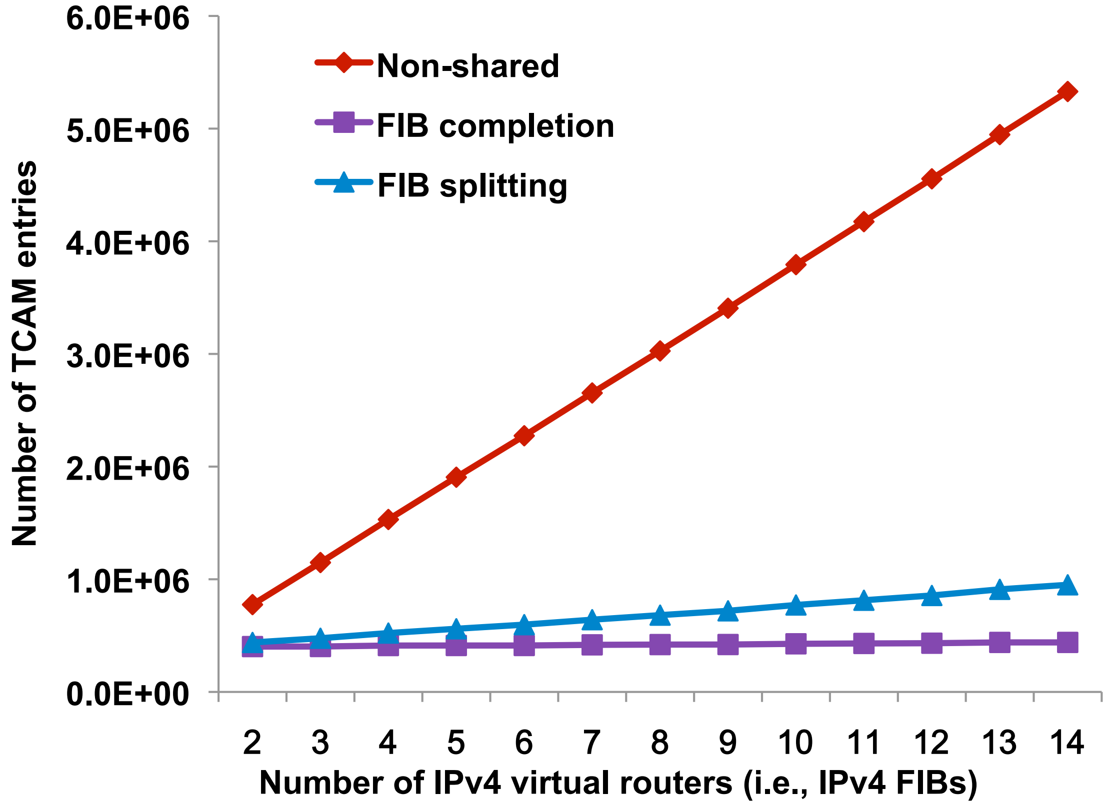
(c) Insert prefix <01*, A7>

(d) Insert prefix <1000*, B7>

Performance evaluation

- Routing tables
  - 14 full routing tables from core routers
- Evaluation
  - TCAM size
  - SRAM size
  - Total cost of the system
  - Lookup and update performance

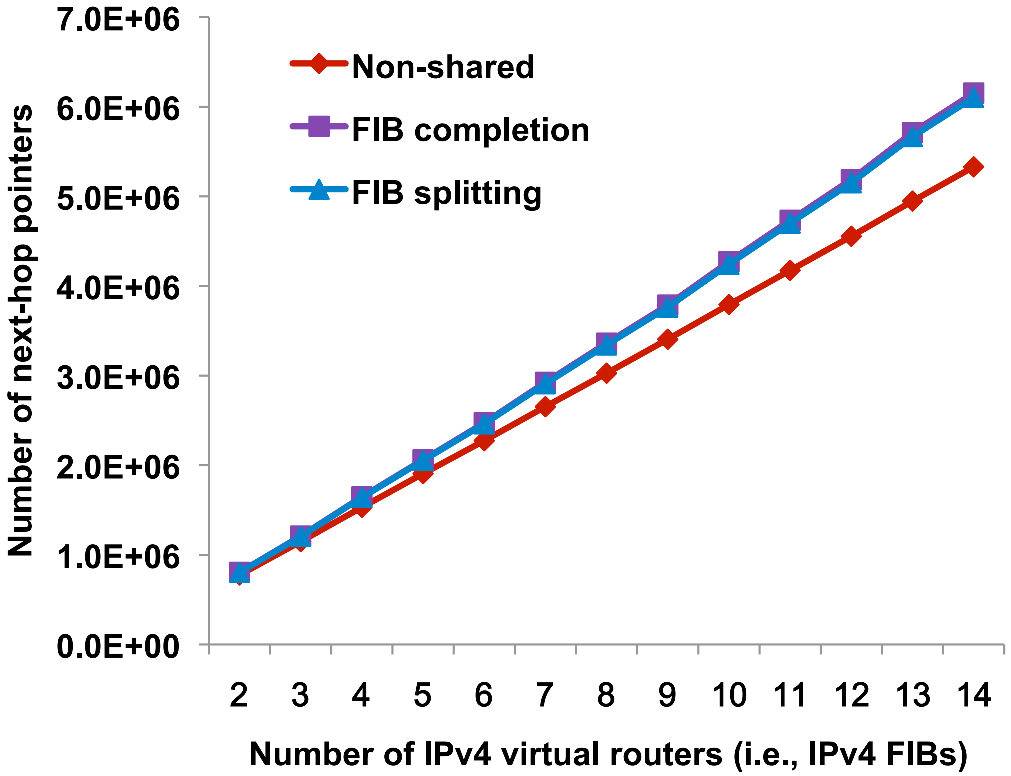
TCAM size



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

Non-shared:	5 M TCAM entries
FIB completion:	429 K TCAM entries
FIB splitting:	928 K TCAM entries

SRAM size



For 14 FIBs:	
Non-shared:	40.7 Mb
FIB completion:	46.9 Mb
FIB splitting:	46.6 Mb

Total cost of the system

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
FIB splitting	3	2	$1341

Lookup & update performance

Table 5. Worst-case lookup 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

Conclusions

- Main contributions
  - The first work to exploit the possibility of using TCAMs to build scalable virtual routers
  - Merged data structure and merging principle for TCAMs
  - Two different approaches
    - 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
  - Quantify the actual update overhead of our approaches based on realistic update workloads

Thank you!