Scalable Name-Based Packet Forwarding: From Millions to Billions

Tian Song, songtian@bit.edu.cn, Beijing Institute of Technology

Haowei Yuan, Patrick Crowley, Washington University

Beichuan Zhang, The University of Arizona

1

A longest-prefix-matching (LPM) algorithm:

Built on binary Patricia trie

Billions

Millions

FIB Size

A LPC algorithm: 3

Built on dual Patricia tries

Speculative Data Plane:

Providing longest prefix classification (LPC)

2

IP vs. Name-based Forwarding

	IP Forwarding	Name Forwarding
Behavior	LPM	LPM
Prefix	IP, less than 4 Bytes	unbounded length
Scheme	4-byte string / word	Hierarchy & flat
FIB Size	$O(10^5)$ [~ 500 K]	$O(10^8)$
Performance	wire speed	wire speed
Memory	SRAM/TCAM	DRAM (mainly)

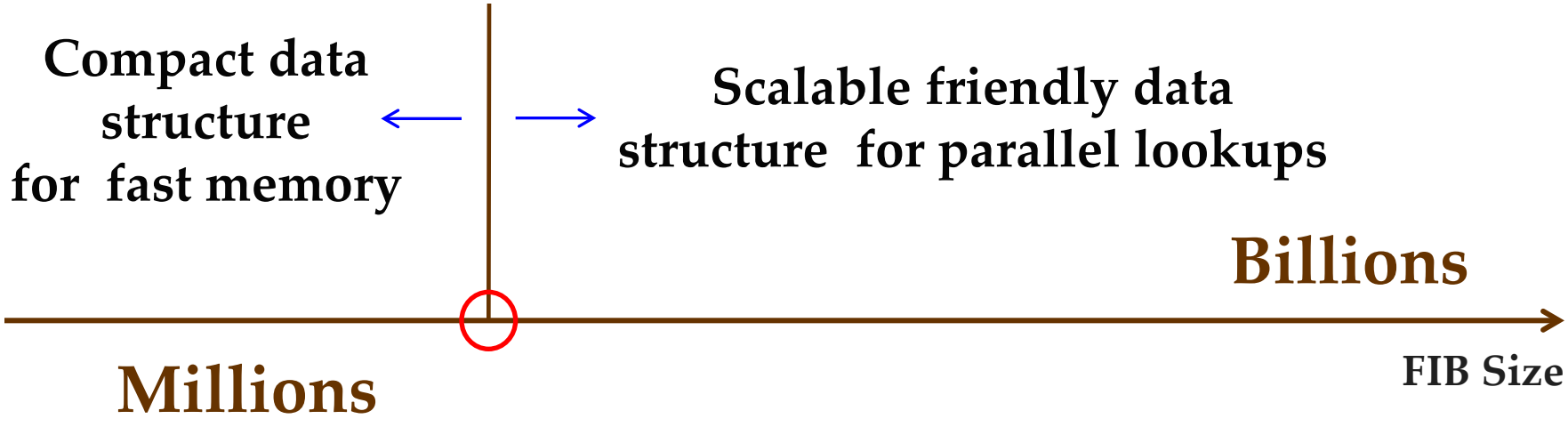
Challenges

DRAM	50 ns read latency / access	## GiB
SRAM	0.47 ns / access; # access / lookup	<135 MiB
TCAM	2.7 ns / lookup	<10 MiB

- The large FIB with unbounded-length names requires a large amount of memory, cross the boundary of the range of SRAM/TCAM.
- i.e. 100 M name prefixes, w/ avg. length 32 B 3.2 GB is required for directly storing prefixes.

Challenges

- A scalable and fast forwarding solution



1

A longest-prefix-matching (LPM) algorithm:

Built on binary Patricia trie

Billions

Millions

FIB Size

A LPC algorithm:

Built on dual Patricia Tries

Speculative Data Plane:

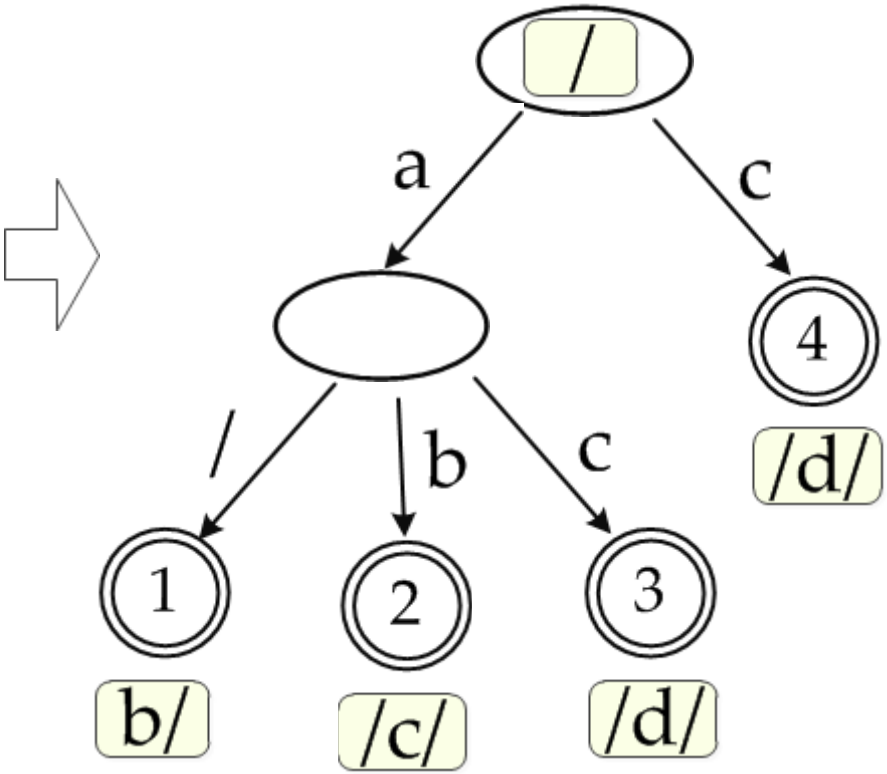
Providing longest prefix Classification (LPC)

Compact Data Structure

- Designed for the scenario of a few million prefixes
- SRAM is considered, which has about 135 MiB
- Avoid assumptions on naming schemes.
- Three potential directions:
  - Hash table –based solutions
  - Component-encoding –based solutions
  - Trie –based solutions

Patricia Trie

prefix	port
/a/b	1
/ab/c	2
/ac/d	3
/c/d	4

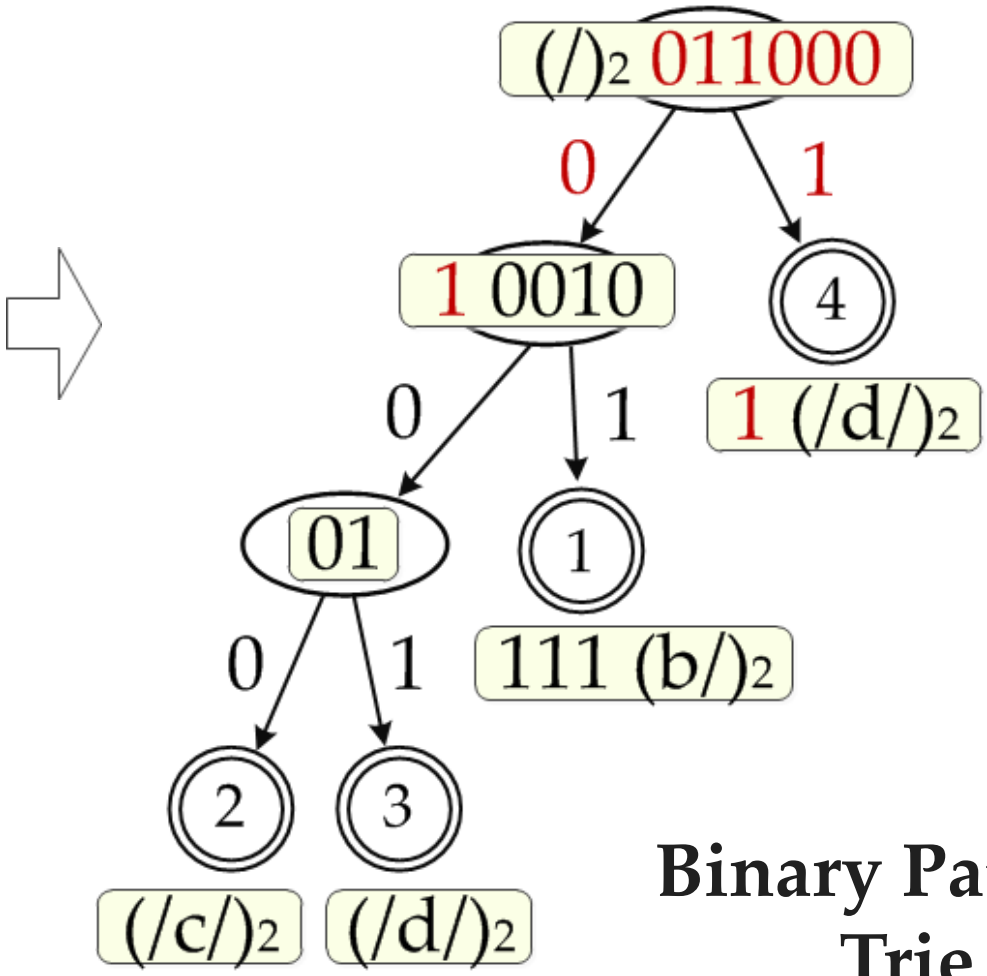


Patricia Trie

is a sub-string.

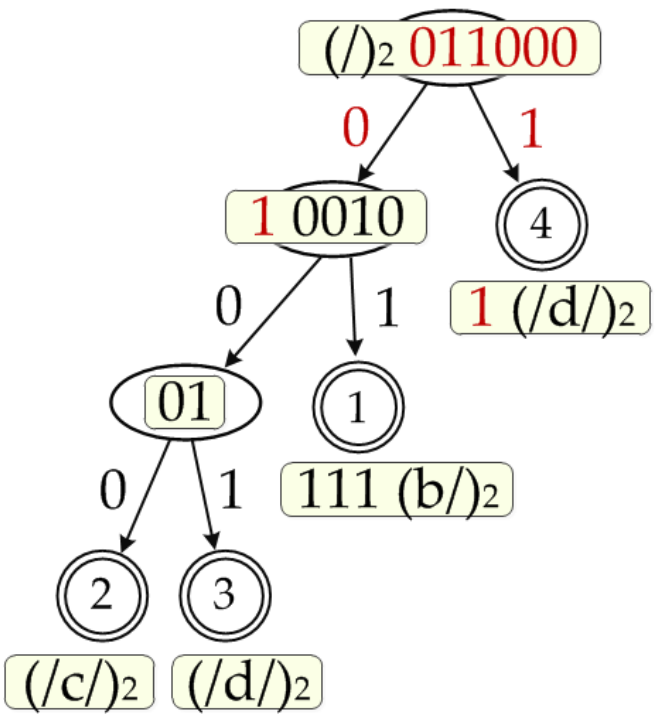
Binary Patricia Trie

prefix	port
/a/b	1
/ab/c	2
/ac/d	3
/c/d	4



Binary Patricia Trie

Binary Patricia-based LPM



- Binary representative is used instead of components.
- Prefixes are decoupled into tokens along the search path.
- Full binary tree can easily be optimized in memory layout.

also, **Tokenized binary Patricia**

Binary Patricia-based LPM

data set	# of keys	avg.len
Alexa 1	999,973	15.5 B
Dmoz 2	3,711,540	27.3 B
Gen100	100 M	37.6 B
Gen300	300 M	37.6 B
Gen1000	1 G	37.6 B

- Comparison Results

for real sets

Method Names	Alexa Mem (1)	Dmoz Mem (2)	Performance
Compressed Trie [25]	N/A	400.8 MiB	0.15 MSPS
NCE [18]	79.64MiB	272.27 MiB	0.91 MSPS
MATA [31](GPU)	N/A	197.6 MiB	63.52 MSPS
Hash [24](GPU)	153 MiB	587 MiB	8.8 MSPS
Binary Patrica (SRAM)	13.56 MiB	64.80 MiB	142 MSPS

~ 3x memory efficiency

A longest-prefix-matching (LPM) algorithm:

Built on binary Patricia trie

Billions

Millions

FIB Size

A LPC algorithm:

Built on dual Patricia Tries

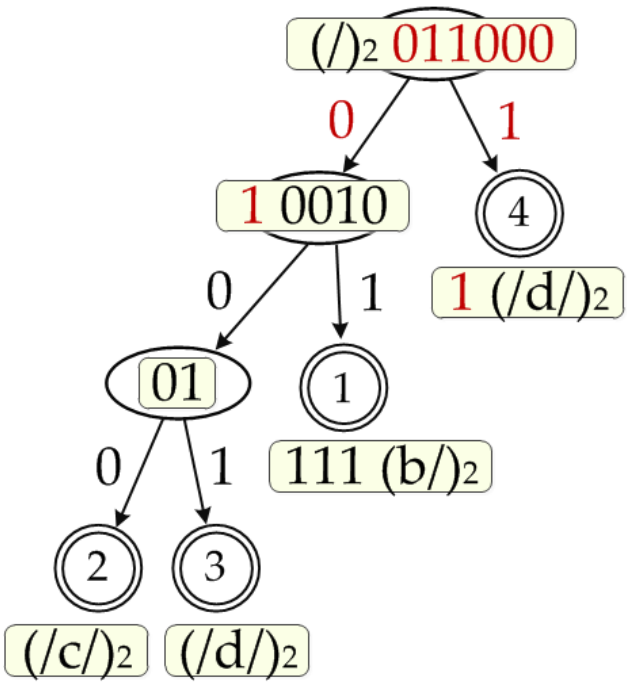
Speculative Data Plane:

Providing longest prefix

Classification (LPC)

2

Tokenized Binary Patricia



- Memory Composition

Memory =

Trie Memory + Token Memory

Trie: information differences

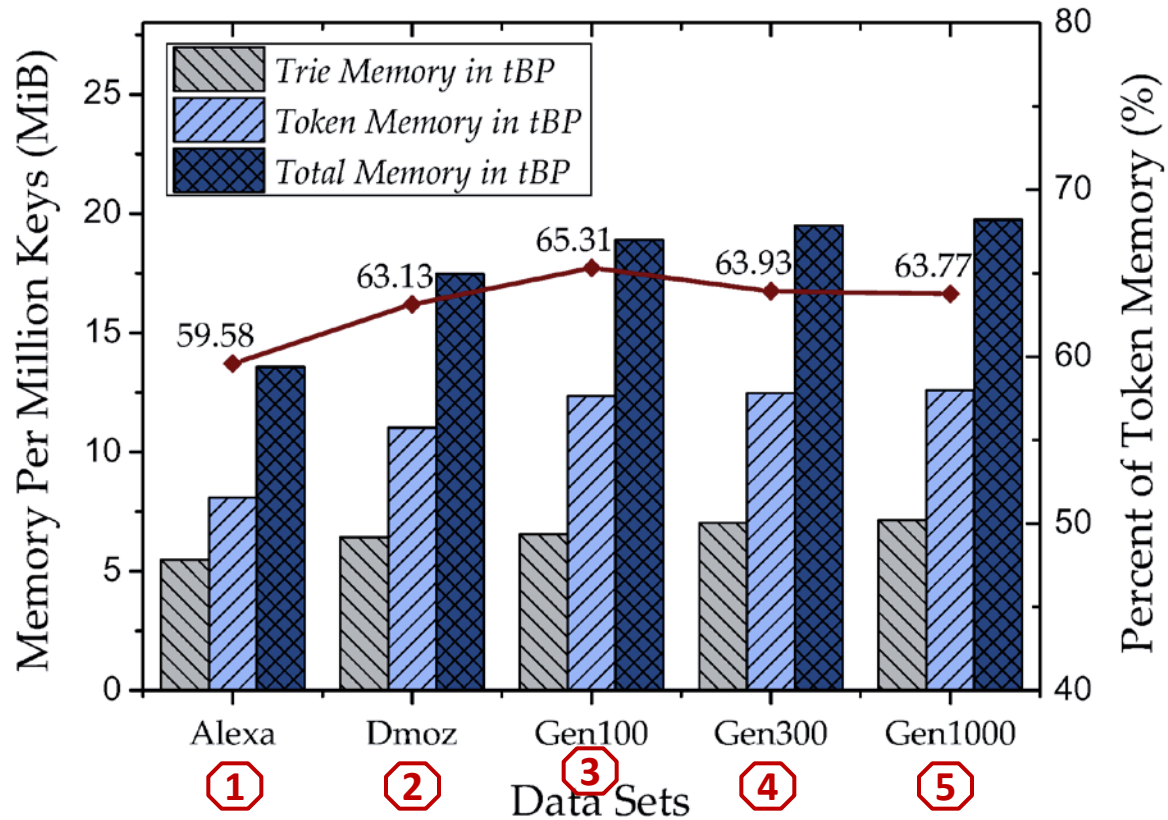
Token: prefix-specific verification

Tokenized Binary Patricia

- Results

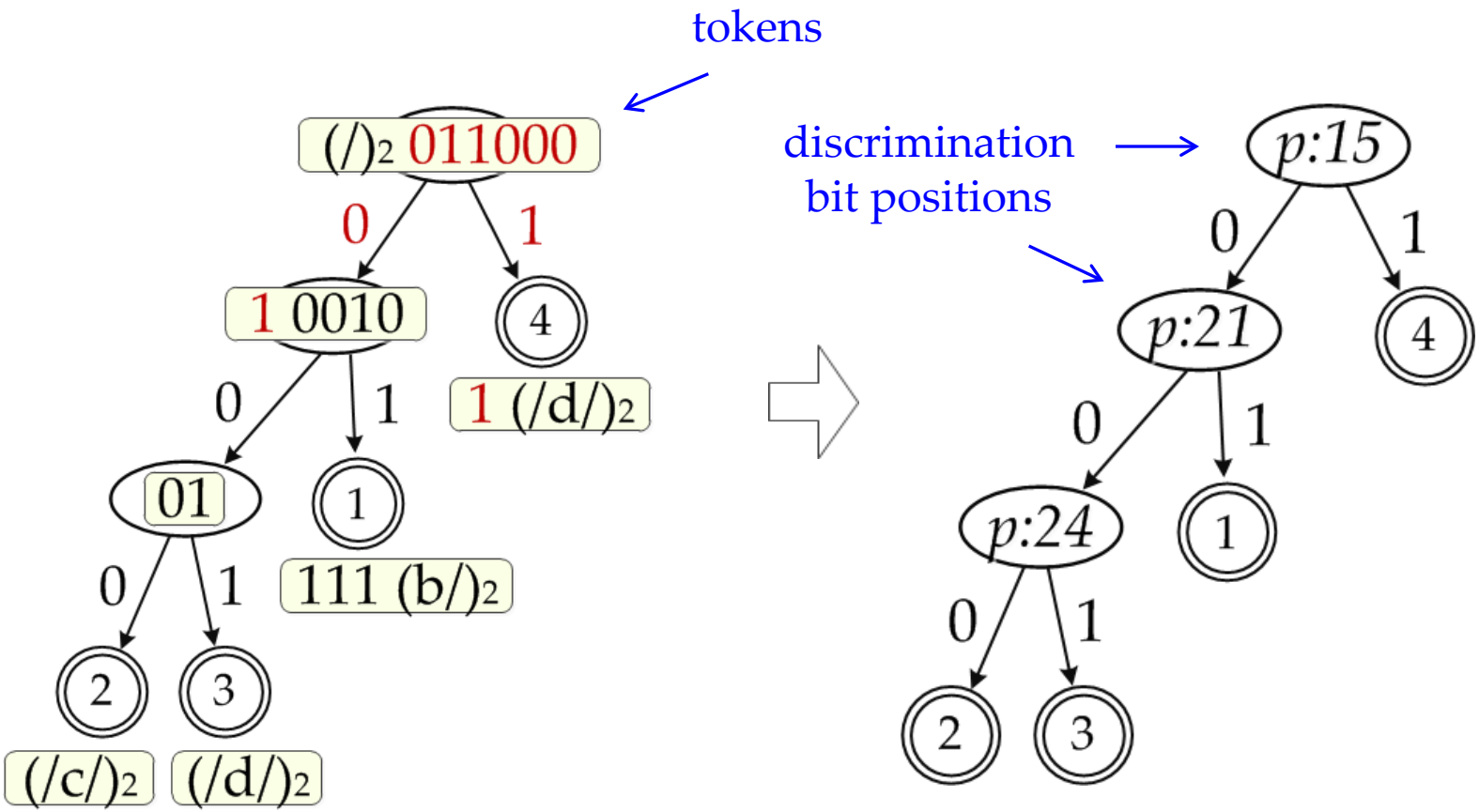
1 M to 1 G URL names

data set	# of keys	avg.len
Alexa (1)	999,973	15.5 B
Dmoz (2)	3,711,540	27.3 B
Gen100 (3)	100 M	37.6 B
Gen300 (4)	300 M	37.6 B
Gen1000 (5)	1 G	37.6 B



Tokens contribute more to memory in terms of scalability.

Longest Prefix Classification



Longest Prefix Matching **Longest Prefix Classification**

LPM = LPC + Verification

Speculative Forwarding

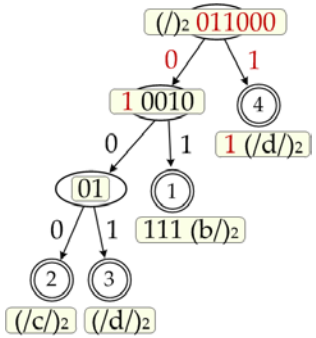
Question:

How to make name-based packet be correctly forwarded by using LPC?

Solution:

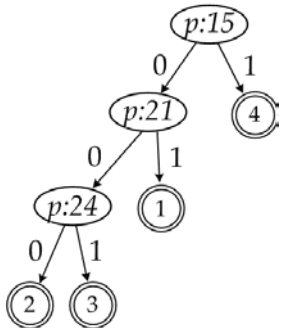
Speculative forwarding is presented, which is defined as a forwarding policy that relays packets by LPC instead of LPM.

Forwarding Behaviors



pkt ② w/ a name: amazon.com/icn

pkt ① w/ a name: facebook.com/icn



Conventional Forwarding

Speculative Forwarding

prefix	*port*
google.com	1
facebook.com	2
youtube.com	3
yahoo.com	4
baidu.com	5

FIB

Drops!

Port: 1 2 3 4 5

① **to *port 2*** ② **is *dropped***

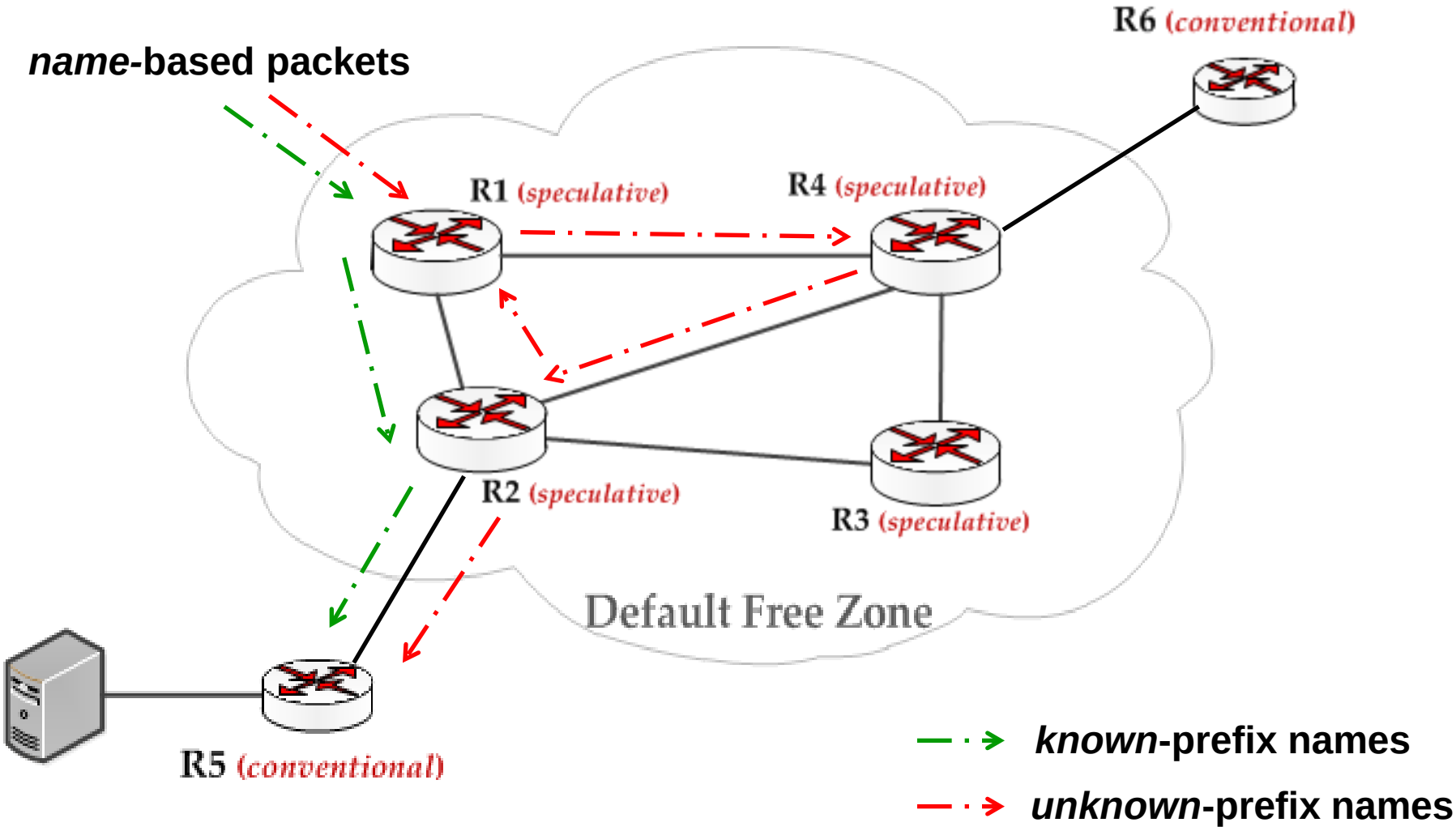
No drops!

Port: 1 2 3 4 5

① **to *port 2*** ② **to *port 5***

LPC drops no packets, so no default path exists.

Speculative Data Plane



Speculative Data Plane

- **Loop Handling for unknown-prefixes:**
  - **NDN**: stateful data plane, loop free in nature
  - Restricted TTL in speculative forwarding
  - Quick feedback (NACK) to remove in-path overhead
  - Other approaches can be applied…
- **Practicability**
  - DFZ routers are performance-critical. LPC helps.
  - Edge routers are function variety. LPM guarantees.

A longest-prefix-matching (LPM) algorithm:

Built on binary Patricia trie

Billions

Millions

FIB Size

A LPC algorithm: 3

Built on dual Patricia Tries

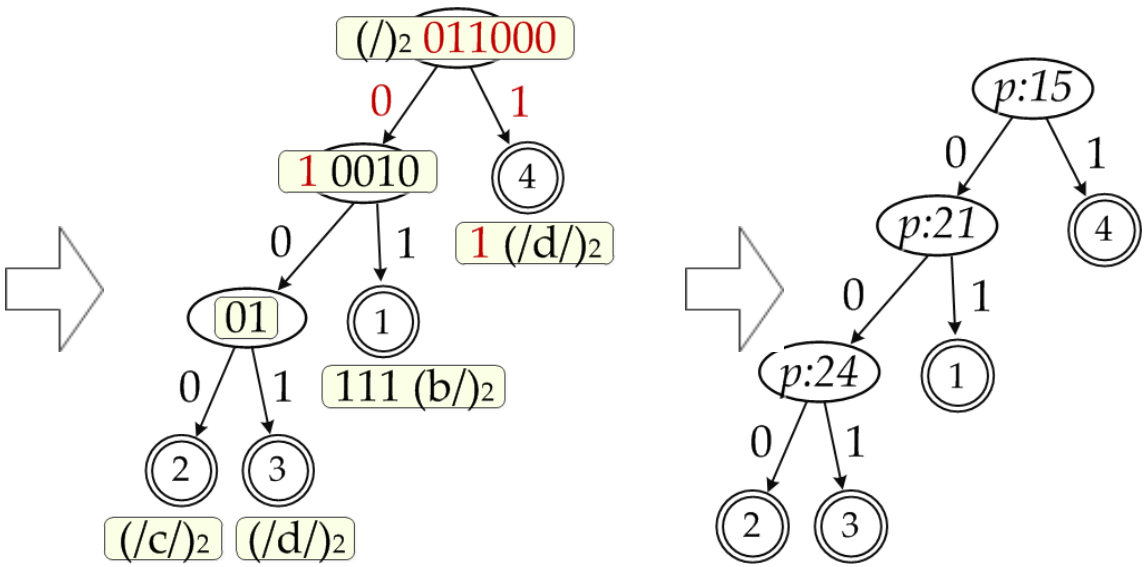
Speculative Data Plane:

Providing longest prefix

Classification (LPC)

Speculative Binary Patricia

prefix	port
/a/b	1
/ab/c	2
/ac/d	3
/c/d	4



Speculative Patricia = Tokenized Patricia - Tokens

Speculative Patricia only supports classification instead of LPC.

For proper prefixes, speculative Patricia cannot distinguish them.

i.e. /a/b is the proper prefix of /a/b/c.

Dual Binary Patricia

FIB

google.com/	1
google.com/us/	2
google.com/us/idn/	3
youtube.com/	4
yahoo.com/	5

Proper Prefix Subset (*P*)

google.com/us/	2
google.com/us/ndn/	3

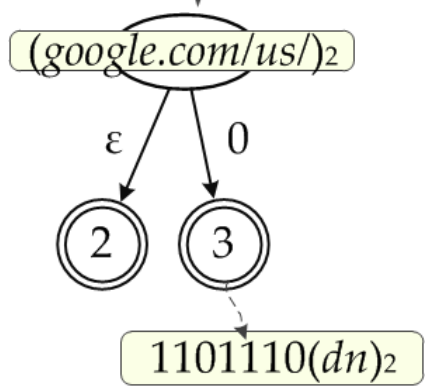
Flat Name Subset (*F*)

google.com/	1
youtube.com/	4
yahoo.com/	5

Dual Patricia

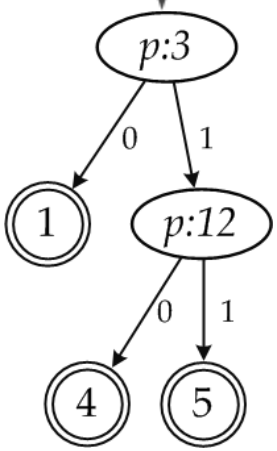
(*DuBP*)

=



***Tokenized* Patricia**

+



***Speculative* Patricia**

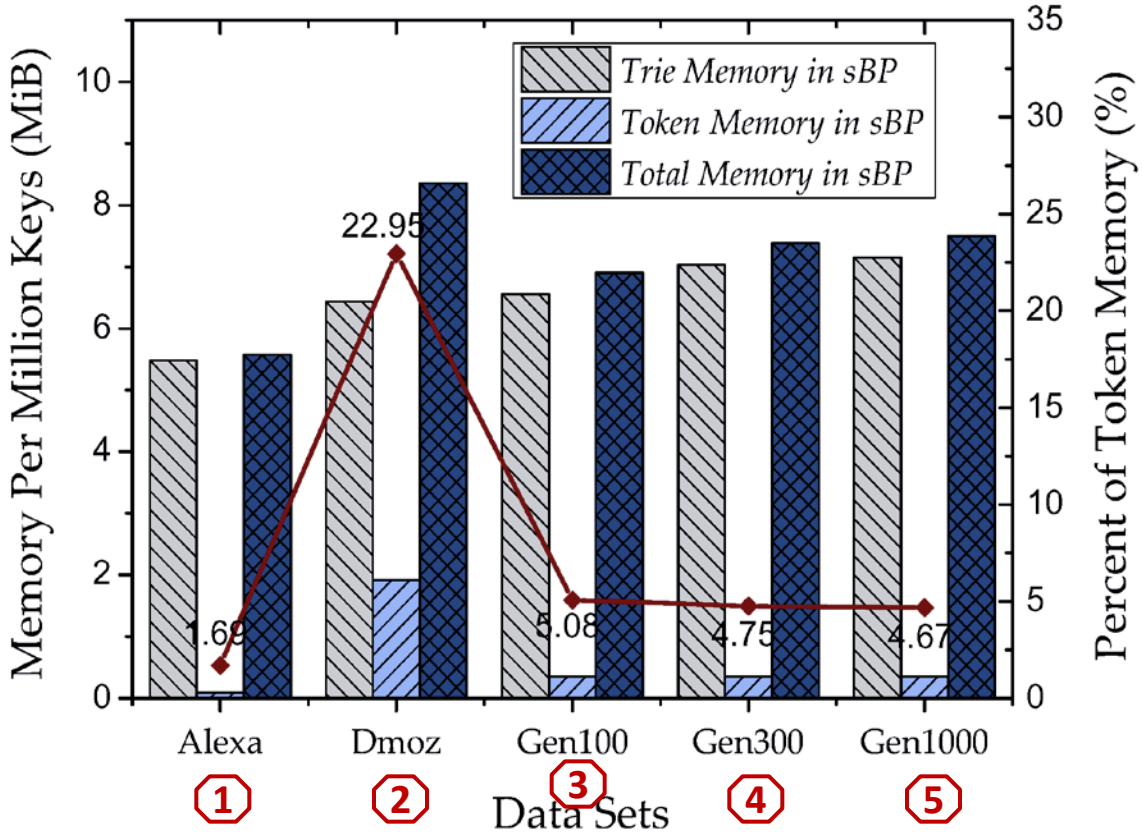
Dual Patricia supports longest prefix classification.

Dual Binary Patricia

- Results

1 M to 1 G URL names

data set	# of keys	% of p.p.
Alexa 1	999,973	1.03
Dmoz 2	3,711,540	3.85
Gen100 3	100 M	5-50/+5
Gen300 4	300 M	5-50/+5
Gen1000 5	1 G	5-50/+5



Dual Patricia is scale mainly to the size of FIB.

Dual Binary Patricia

- Results

Data Set	Tokenized Patricia(*tBP*)			
	Trie	Token	Total	Devices
Alexa (MiB)	5.48	8.08	**13.56**	SRAM
Dmoz (MiB)	23.89	40.91	**64.80**	SRAM
Gen100 (GiB)	0.64	1.21	1.85	DRAM
Gen300 (GiB)	2.06	3.62	5.68	DRAM
Gen1000 (GiB)	6.98	12.05	**19.03**	DRAM

from LPM to LPC

~ 2.6x memory efficiency

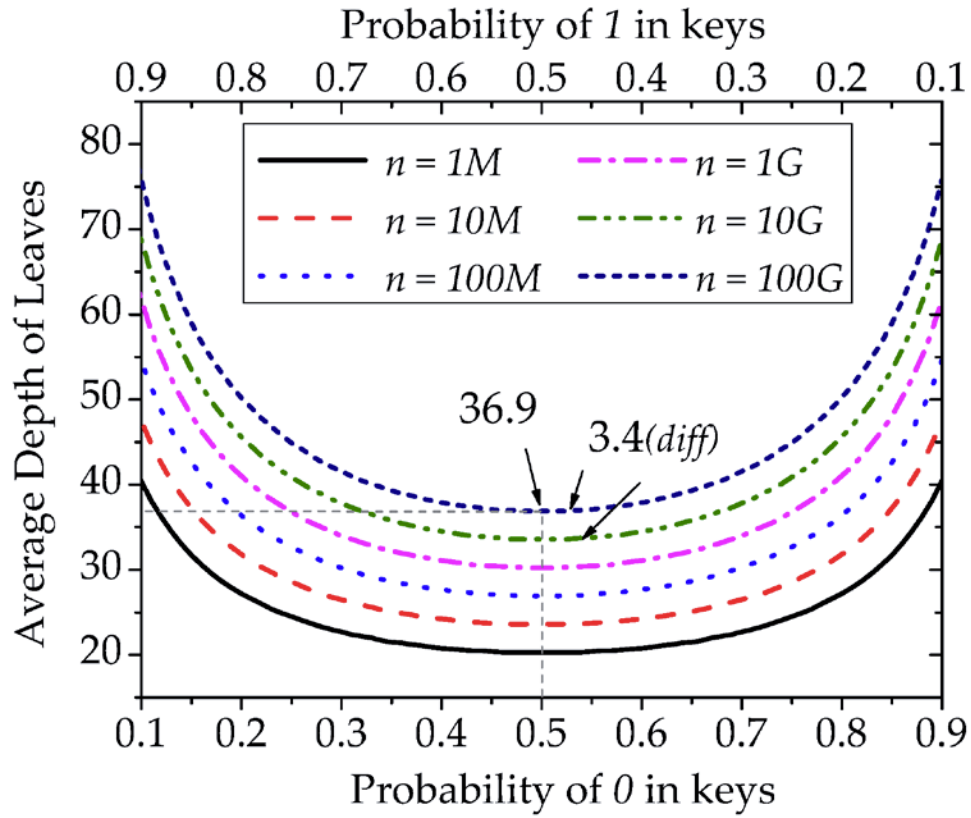
Data Set	Dual Patricia(*DuBP*)			
	Trie	Token	Total	Devices
Alexa (MiB)	5.48	0.10	**5.58**	SRAM
Dmoz (MiB)	23.89	7.12	**31.01**	SRAM
Gen100 (GiB)	0.64	0.034	0.67	DRAM
Gen300 (GiB)	2.06	0.103	2.16	DRAM
Gen1000 (GiB)	6.98	0.342	**7.32**	DRAM

data set	# of keys	avg.len
Alexa	999,973	15.5 B
Dmoz	3,711,540	27.3 B
Gen100	100 M	37.6 B
Gen300	300 M	37.6 B
Gen1000	1 G	37.6 B

Discussions

- Scalability from millions to billions

Average Depth in Patricia



Given :

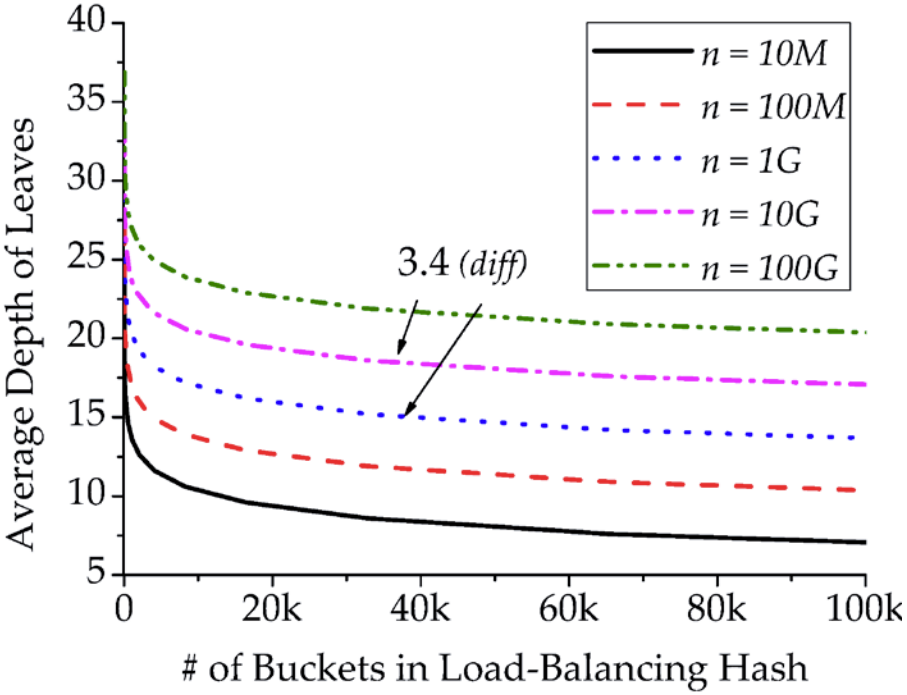
1 M to 100 G FIB Size

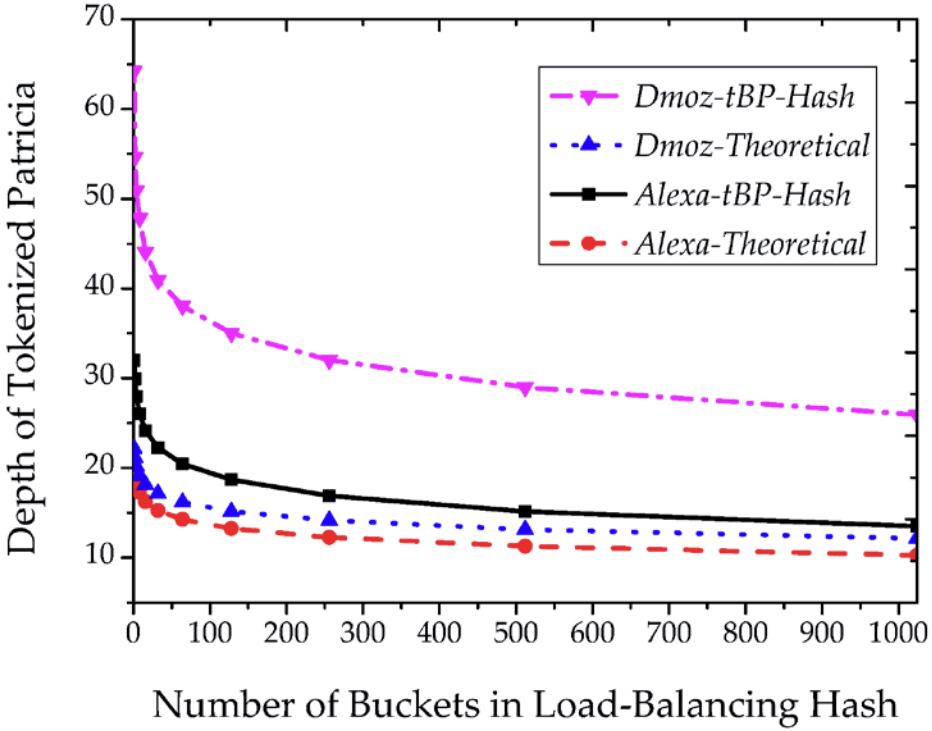
3.4 more depths for 10x size

Patricia trie scales humbly in its depth.

Discussions

- Trie lookup speed: also related to trie depths





A load-balancing hash with hundreds of buckets can reduce depth to 10 to 15.

Therefore, SRAM / DRAM can be well optimized.

Summary

1 **A longest-prefix-matching (LPM) algorithm:**

Built on binary Patricia trie

~ 3x memory efficiency than other solutions for SRAM and DRAM

Billions

Millions

FIB Size

A LPC algorithm: 3

lt on dual Patricia Tries

2.6x memory efficiency than 1
142 MSPS on SRAM (284 Gbps)
20 MSPS on DRAM (62 Gbps)

Novel data plane for fast forwarding

Speculative Data Plane: 2

Providing longest prefix Classification (LPC)

Q & A!