

Vasilis Georgiou

November 2024

## 1 Datasets

Datasets are split in REAL or SYNTHETIC and are based on the the source of the routing table:

- REAL-Tier-1-A: real core backbone router in a global tier-1 ISP
- REAL-Tier-1-B: national backbone router in research and educational network of WIDE Project.
- SYN1: each procedure that is no longer than /24 and /16 is split into two and four prefixes
- SYN2: Each prefix that is no longer than /24, /20 and /16 is split into two, four and eight prefixes

**Notation:**

- binary radix depth: longest prefix matching

## 2 Traffic Patterns

Take the following traffic patterns into consideration:

- random:  $2^{32}$  random traffic patterns (generated by xorshift).
  - overhead for generating the data is included in the measurements, but it is small.
- sequential:
  - Generates in the range 0.0.0.0 to 255.255.255.255
- repeated
  - Similar to random but each random lookup is repeated 16 times
- real-trace
  - real traffic trace

### 3 Using the different libraries

- modified\_poptrie
  - need to copy the test data files to modified\_poptrie/build/tests
  - last test doesn't pass
- modified\_radix\_tree
- modified\_tree\_bitmap
  - rm\_test\_v6 need to pass the input file here to measure runtimes.
  - use the runtime information and memory usage to compute the lookup rate
- modified\_sail
  - uncomment runtime measurement commands in function sailPerformanceTest()
  - QueryPerformanceFrequency() function doesn't exist
  - QueryPerformanceCounter() doesn't exist

#### 3.1 Datasets included in libraries

- modified\_poptrie/tests
  - linux-rib.20141217.0000-p46.txt
  - linux-rib.20141217.0000-p52.txt
  - linux-rib-ipv6.20141225.0000-p69.txt
  - linux-update.20141217.0000-p52.txt

#### 3.2 Comments on preallocation of memory for data-structures

- It makes sense to preallocate, because the data structures will be initialized once and then remain as is.
- lookup rate = number of lookups / total runtime

Configuration	$s$	# inodes	# leaves	Mem (MiB)	Init (s)	Rate (Mlps)	CPU cycles
Radix	-						
Poptrie	2	36,412	63,527	7.575	2.16	71.91	45
	16	14,664	56,367		1.79	285,7	
	18	14,664	56,367	6.12	1.80	316.8	

Table 1: The compilation time, the number of nodes, the memory footprint, and the lookup rate for random with direct pointing ( $s = 0, 16, 18$ )

Configuration	$s$	total memory	routes	trie memory	loads	stores
Poptrie	2	94.6	55.5	39.1	7.450	0.124
	16		routes	trie memory	loads	stores
	18		routes	trie memory	loads	stores

Table 2: The total allocated memory, memory used for the routing table, memory footprint, load accesses, store accesses for poptrie with leaf compression, direct pointing and  $s = 0, 16, 18$  (MiB)


### 3.3 Tables

**characteristics:**

- poptrie-basic-s
- poptrie-leafvec-s
- poptrie-s (direct pointing)
- #inodes
- #leaves
- memory footprint
- compilation time (we measure this because we reconstruct the tree. Does this include rebuilding the tree?)

### 3.4 Poptrie specification

- FIB: forwarding information base
- RIB: routing information base

 figures/massif\_poptrie\_s2.png