Power-Saving Hybrid CAMs for Parallel IP lookups

Heeyeol Yu and Rabi Mahapatra
Texas A&M University
Email: {hyyu,rabi}@cs.tamu.edu

Uichin Lee
University of California, Los Angeles
Email: uclee@cs.ucla.edu

Abstract— **IP lookup with the longest prefix match is a core function of Internet routers. Partitioned Ternary Content Addressable Memory (TCAM)-based search engines have been widely used for parallel lookups despite power inefficiency. In this paper, to achieve a higher throughput and power-efficient IP lookup, we introduce hybrid CAM (HCAM) architecture with SRAM. In our approach, we break a prefix into a collapsed prefix in CAM and a stride in SRAM. This prefix collapse reduces the number of prefixes that results in reduced memory usage by a factor of 2.8. High throughput is achieved by storing the collapsed prefixes in partitioned CAMs. Our record for 2 BGP tables shows that an HCAM saves 3.6 times energy consumption compared to the TCAM based implementation.**

I. INTRODUCTION

IP lookup is one of the key issues in a critical data path for high-speed routers, and its challenges arise from the following: 1) The IP prefix length distribution varies from 8 to 32 and the incoming packet does not carry the prefix length information for the IP lookup. 2) One IP address may match multiple prefixes in a forwarding table and the longest matching prefix should be chosen. 3) In addition to the IP lookup complexity, the number of hosts is tripling every two years. 4) It has been reported that the traffic of the Internet is doubling every two years by the Moore's law of data traffic.

To cope up with these challenges, the literatures on IP lookup describe schemes involving three major techniques, trie-based, hash-based, and Ternary Content Addressable Memory (TCAM). Because of an irregular prefix distribution in a trie's tree structure, an imbalanced memory access in trie pipeline hinders a fast IP lookup. Such hinderness also intrigues Circular-Adaptive-Monotonic pipeline (CAMP) [1] although its throughput is 0.8. Hash-based schemes like a Bloomier filter-based HT (BFHT) [2] provide a flat memory architecture. However, a BFHT undergoes setup failure and $O(n \log n)$ setup and update complexities, where n is the number of prefixes. Also, it does not provide a high throughput.

Unlike trie and hash approaches, TCAMs have become the de facto industrial standard solution for high-speed IP lookup. More than 6 million TCAM devices were deployed worldwide in 2004. Despite TCAMs' popularity and simplicity, TCAMs have their own limitations as follows: 1) Throughput: parallel searches in all prefixes are made in one clock cycle for a single lookup, so that the throughput is simple 1 as in Table I. 2) Power: such an one-cycle lookup causes at most 150 times more power consumption than a SRAM-based trie and a hash as shown in Table II. Thus, reducing TCAM power usage is a paramount goal for a deterministic TCAM lookup.

schemes				TCAM	CAM	SRAM
Trie	$O(W)$ [†]	Power [‡]		≈15	≈1	≈0.1
Hash	$O(1)$	Cell°		16	8	6
TCAM	1					

TABLE I	TABLE II
LOOKUP COMPLEXITY	H/W FEATURES

[†] *W*: the number of IP address bits

[‡] Watts unit

° # of transistors per bit

A high TCAM throughput has been achieved through a partitioning technique [3, 4]. Its principle with pipelining lies in a parallel architecture that fulfills multiple lookups per clock cycle. Likewise, authors in [5] claim that partitioning a trie and maps subtries to pipelines is a *NP-complete* problem and propose a SRAM-based parallel scheme with a heuristic algorithm for a high throughput. However, these kinds of approaches suffer high power consumption and complicated mapping algorithm complexity, respectively.

In this paper, we propose a hybrid CAM (HCAM) IP lookup architecture for high throughput and power efficiency. A prefix collapse (PC), breaking a prefix into a collapsed prefix and a stride, reduces the number of prefixes as opposed to the prefixes expansion. In such prefix collapse, the collapsed prefixes (CPs) can be put in a deterministic lookup-capable CAM to demonstrate further hardware efficiencies on power and a cell than a TCAM can as shown in Table II. A complete prefix match beyond the CP match is made through a stride tree bitmap (STB) saved in SRAM. Also, the CAM for the same-length CPs can be partitioned into CAM blocks to provide multiple lookups on the CPs per clock cycle.

II. HCAM ARCHITECTURE OVERVIEW

Our HCAM scheme has two major features; a PC and a STB with help of a Bloom filter lookup distributor
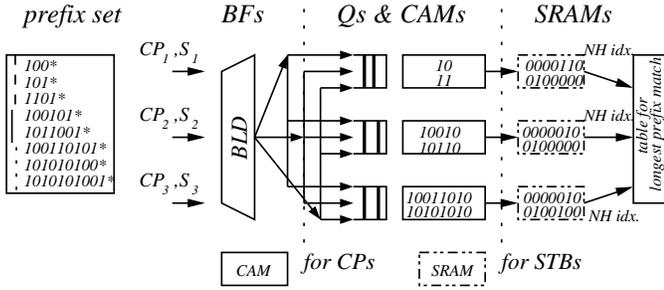
Fig. 1. HCAM architecture with 8 prefixes. 3 CAMs are used for 8 CPs. 3 pairs of packets' CPs and stride Ss are fed into a BLD.

(BLD) [6]. These features are working in both parallel and pipeline for the same length CPs and their STB as shown in Fig. 1. In the figure, 3 CPs are fed into a BLD together, and the BLD distributes the CPs to their CAM blocks through queues. Once a CAM block entry is matched with a CP, an SRAM block entry at the same index indicates this CP's STB for stride match. Thus, the prefix match is achieved by performing CAM and STB matches. As to completing the longest prefix match, a table records matched lookups among CAMs. All SRAM matches for a lookup are recorded, and if a match is found the longest prefix match, the lookup is to find a next hop at last.

A. Prefix Transformation in CAM and SRAM

Since a CAM does not support a prefix match, a stride match made after a CAM match is necessary. Given a CP, there are $2^{s+1}-1$ possible prefixes at stride s, and they can be presented at a tree bitmap. Fig. 2 a) shows three prefix strides at stride $s=3$ and a stride tree for them. In a stride tree, a node is marked as '1' when there is a corresponding prefix stride. Thus, when we scan nodes' bits with the horizontal order followed by the vertical order, a STB for three prefix strides becomes grouped in (00100000,0010,01,0) of 15 bits.
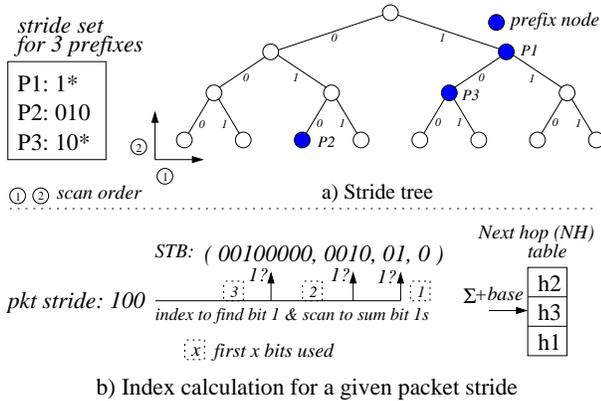

a) Stride tree

b) Index calculation for a given packet stride

Fig. 2. A stride tree for 3 prefix strides and an index calculation

Given a STB for the stride s, grouped bits are scanned

while the number of bits of value 1 is counted, and when a bit indexed by the most significant bits in a stride is 1, the counting stops. Then the summed number of 1-bits, $\sum$, becomes a relative index to an NH table. Fig. 2 b) shows such an index calculation in the STB for the packet stride 100. Once a CAM block match happens, the match's index in the CAM block is used to access a STB in the corresponding SRAM block.

III. EXPERIMENTAL RESULTS ON POWER AND MEMORY

Although we observed that the HCAM throughput is proportional to the number of HCAM blocks as in [5], we do not show the result data due to page limit.
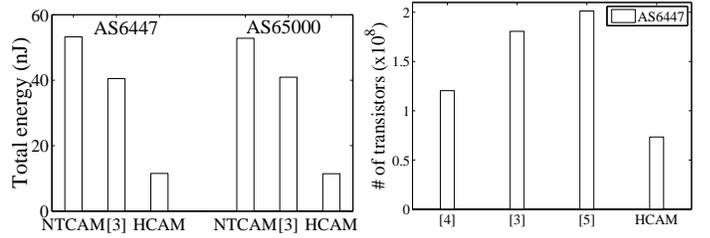
(a) Total energy per clock cycle   (b) Total number of transistors

Fig. 3. Power and memory measurements

By using the TCAM and CAM modeling tools, we measured the total energy in one clock and memory capacity for three approaches: a naive TCAM (NT-CAM), schemes in [3], and our HCAM. Such energy consumptions are shown in Fig. 3(a) for two routing tables (AS6447 and AS65000). An NTCAM in the figure provides only one lookup with the entire prefixes while a Ultra TCAM (UTCAM) [3] and an HCAM can provide multiple lookups with TCAM or CAM blocks. On average, a UTCAM and an NTCAM use 3.6 and 4.6 times the energy of an HCAM, respectively.

Fig. 3(b) shows the memory comparison among other schemes [3–5] and our HCAM. We consider the number of transistors to account for the different TCAM and CAM hardware features. In contrast to [5]' scheme, our HCAM scheme uses 2.8 times less memory while providing the similar throughput amount.

REFERENCES

[1] S. Kumar, M. Becchi, P. Crowley, and J. Turner, "CAMP: Fast and Efficient IP Lookup Architecture," in *ANCS '06*.
[2] J. Hasan *et al.*, "Chisel: A Storage-Efficient, Collision-free Hash-based Network Processing Architecture," in *ISCA '06*.
[3] K. Zheng *et al.*, "An Ultra High Throughput and Power Efficient TCAM-Based IP Lookup Engine," in *INFOCOM '04*.
[4] J. Akhbarizadeh *et al.*, "A TCAM-Based Parallel Architecture for High-Speed Packet Forwarding," *IEEE Trans. Comput.*, 2007.
[5] W. Jiang *et al.*, "Beyond TCAMs: An SRAM-based Parallel Multi-Pipeline Architecture for IP Lookup," in *INFOCOM '08*.
[6] H. Song, J. Turner, and S. Dharmapurikar, "Packet Classification Using Coarse-grained Tuple Spaces," in *ANCS '06*.