High-Performance TCAM-Based IP Lookup Engines

Hui Yu[†], Jing Chen[‡], Jianping Wang[§] and S.Q. Zheng[† ‡ *]
[†] Computer Engineering Program, University of Texas at Dallas, Richardson, TX 75083
[‡] Telecommunications Engineering Program, University of Texas at Dallas, Richardson, TX 75083
[§] Department of Computer Science, City University of Hong Kong, Hong Kong
[*] Department of Computer Science, University of Texas at Dallas, Richardson, TX 75083

***Abstract*— IP lookup is a key packet processing task in IP routers. To achieve high performance, it has been proposed to use TCAMs to implement IP-lookup accelerators. In this paper, in addition to the state-of-art MSMB-LPT scheme for TCAM-based IP lookup, M-MSMB-LPT and MSMB-LPT-I schemes are proposed and studied. Using realistic Internet packet traces, it is shown that MSMB-LPT, M-MSMB-LPT and MSMB-LPT-I schemes with different parameters constitute a solution space for constructing a wide-range of high-performance TCAM-based IP lookup engines. Under cost, performance and scalability constraints best IP lookup engine can be selected from this solution space.**

Keywords: IP router, IP lookup, IP packet forwarding, IP packet processing, longest prefix match, routing table, TCAM.

I. Introduction

Longest prefix matching (LPM) is a fundamental problem in Internet routers. Classless Inter-Domain Routing requires Internet router to search the longest matching prefix among variable-length IP address prefixes and retrieve the corresponding forwarding information for each packet traversing the router. This task, which is referred to as IP lookup, is a performance bottleneck in Internet routers. Intensive efforts have been devoted to IP lookup performance.

Various software IP lookup techniques have been proposed. These techniques were based on efficient data structures, such as tries, trees, and hashing tables, and related search algorithms. (e.g. [1], [2], [3], [4]). Various hardware lookup engines, such as application-specific integrated circuits (ASICs) coupled with SRAMs and DRAMs, have also been proposed. As recent advances in optical networking technology have pushed data transfer rates in high-speed IP routers to a new level with line-card rates increasing to 40Gb/s, it is critical that all packet processing tasks performed by a router should be able to operate in nanosecond time frames. RAM-based software and hardware lookup engines are considered too slow because their performance depend on the number of memory accesses for each packet.

Hardware-based techniques using content addressable memories (CAMs) allow an input key to be compared against all memory words in parallel. CAM has been considered a major component in high-performance IP lookup engines to match packet transmission rates [5]. For example, ternary content addressable memory (TCAM) has the capability of achieving single clock cycle LPM (IP lookup) operation by storing the "don't-care" state. The major disadvantage of TCAM is its high cost due to its high power consumption and high circuit density. This is why one TCAM-based routing table is shared by multiple packet streams in one line card or by multiple line cards in practice, such as in Cisco 10000 series routers, for cost effectiveness. Panigrahy and Sharma [6] proposed a framework of reconfiguring a TCAM into several independent blocks so that parallel IP lookup is possible. According to this framework, a TCAM-based routing table is implemented by k TCAM blocks (i.e. TCAM chips), which are shared by m key inputs. Any key input can select any TCAM block, and $\min\{m, k\}$ different lookups can be fulfilled in parallel per cycle in the best case. Such a TCAM-based structure is named *Multi-Selector and Multi-Block* (MSMB) scheme. Several general block partitioning methods and block selection schemes were proposed in [6]. Additional block partitioning and management methods were proposed in [7], [8]. In these studies, the emphases of partitioning and management methods are to balance the load of the TCAM blocks, to avoid bias towards any TCAM block, and to reduce TCAM power consumption. These methods require periodic table reconstruction and rearrangement, leading to inefficient use of TCAM resource and complicated designs that are not rapidly responsive to traffic changes.

To alleviate the TCAM contention problem caused by traffic bias, the *Multi-Selector and Multi-Block Popular-Prefix Table* (MSMB-PT) scheme based on temporal locality of packet destinations was proposed in [9]. In MSMB-PT, each TCAM selector at input is augmented with a small associative memory, called *Popular-Prefix Table* (PT), caching some of the prefixes recently used by all inputs. Compared with the previously proposed MSMB schemes (e.g. [6]), MSMB-PT scheme achieves improved speedup and throughput with smaller adaptive table management overhead [9]. It was observed in [10] that the PTs, though attached to different selectors, are a performance bottleneck of MSMB-PT in forms of parallel-search-sequential-update and small effective total PT size, which are caused by their shared single content. Based on the self-similar and bursty behavior of Internet traffic, an improved MSMB scheme, the *Multi-Selector and Multi-Block Local Popular-Prefix Table* (MSMB-LPT) scheme, was proposed in [10]. The major difference between MSMB-LPT

978-1-4244-2219-7/08/$25.00 (c)2008 IEEE

and MSMB-PT is that PTs in MSMB-PT capture temporal locality global to all inputs, whose existence is not strongly evident, whereas *Local Popular-Prefix Tables* (LPTs) capture temporal locality of flow which is based on bursty patterns of Internet traffic[11], [12], [13]. design is based on the behavior of Internet traffic. It was shown by simulations that MSMB-LPT schemes improve the performance of MSMB-PT schemes by up to 250%, 80%, 82%, and 71% in speedup, hit ratio, TCAM contention, and TCAM power consumption, respectively. Since the architecture of MSMB-LPT is simpler than that of MSMB-PT, these performance improvements are achieved without introducing additional cost.

The number k of TCAM blocks in an MSMB system cannot be arbitrary. Decreasing k results in increased size, cost and search time of each TCAM block and reduced number of blocks, leading to more contention and less parallelism. If k is too large, then the size of each TCAM block is too small and too many TCAM blocks are needed. Consequently, the associated interconnection and control circuits become too complicated, resulting high cost and performance overhead.

This paper addresses the following questions related to TCAM-based IP lookup engines: How to obtain a TCAM-based IP lookup engine that improves MSMB-LPT without using more hardware resources? How to design TCAM-based IP lookup engines that satisfy given performance requirements? For large number m of inputs, how to design scalable TCAM-based IP lookup engines? How to find tradeoffs among cost, performance and reliability. We answer these questions by presenting two new TCAM-based MSMB schemes: Multiple MSMB-LPTs (M-MSMB-LPT), and MSMB-LPT with interleaved TCAM blocks (MSMB-LPT-I). We show that MSMB-LPT, M-MSMB-LPT, and MSMB-LPT-I schemes give rise to a wide range of design choices for designing high-performance TCAM-based IP lookup engines. Using simulations, we compare the performances of different TCAM-base IP lookup engines.

II. MSMB-LPT SCHEME

A *flow* is a stream of packets that traverse the same route from source to destination. A flow can be generated in many different ways such as transmitting an email, a file, an image, a video or audio script, for which the packets are transmitted as a bursty sequence. For a given IP router R, the packets of flows arrive at the same input of *R* exhibit bias of IP streams to a small set of IP prefixes with respect to that input in a time interval. For any bursty traffic period of an input of *R*, the bias of IP addresses is called the *temporal locality of flows*. The design of the MSMB-LPT scheme was based on this locality of reference.

Figure 1 is the block diagram of the MSMB-LPT scheme. There are m key inputs and k TCAM blocks $TCAM_j$, $1 \leq j \leq k$, which implement a k-way partitioned lookup table. Associated with each key input i there is a *range detector* RD_i. According to the current incoming key of input i, RD_i determines which TCAM block the key belongs to. Also associated with each key input i is a *local popular table* LPT_i, which is an associative memory unit capable of storing prefixes; i.e. it may be a small TCAM by itself. It is used to dynamically store recently referenced IP prefixes requested from input i. When Sel_i receives a search key, LPT_i is looked up. If there is a match in LPT_i, the result is sent to the output. If there is no match in LPT_i, the search key has to be sent to $TCAM_j$ determined by RD_i as a TCAM search request. If at the same time, other RDs (selectors) also try to place requests for $TCAM_j$, contention occurs at $TCAM_j$. CR_j chooses one request according to a priority scheme and passes it to $TCAM_j$. $TCAM_j$ only sends $result_i$ to Sel_i to update LPT_i. MSMB-LPT is created to alleviate TCAM contention problem caused by traffic bias. LPT helps to reduce the number of accesses to the TCAM chips and TCAM contentions. Simulation results of [10] show that the performance of MSMB-LPT is significantly better than MSMB-PT in all aspects.

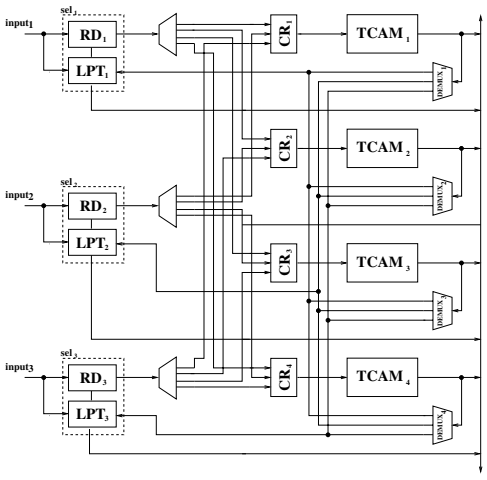

Fig. 1. Block diagram of the MSMB-LPT scheme

III. M-MSMB-LPT AND MSMB-LPT-I SCHEMES

We say that an MSMB-LPT has a configuration (m, n, k) if it has m inputs and k TCAM blocks, and each input is associated with an LPT of size n. Let M be the total number of prefixes to be stored. In an MSMB-LPT scheme of configuration (m, n, k), each TCAM block contains $\frac{M}{k}$ prefixes. The parameters m and k are carefully selected to achieve optimized cost and performance taking the cost and search time of a TCAM chip into consideration. Are there better MSMB schemes for given m and k? We attempt to answer this question.

A. M-MSMB-LPT

For large m, we propose to use w identical copies of MSMB-LPT of configuration (m', n, k), $m' = \frac{m}{w}$, with input $i \cdot m' + j$, $0 \leq i < w$ and $1 \leq j \leq m'$, as the j-th input of the $(i+1)$-th MSMB-LPT. The j-th MSMB-LPT has k CRs and k TCAM blocks, with $CR_{j,u}$ associated to $TCAM_{j,u}$, $1 \leq u \leq k$. The w TCAM blocks $TCAM_{j,u}$,

978-1-4244-2219-7/08/$25.00 (c)2008 IEEE

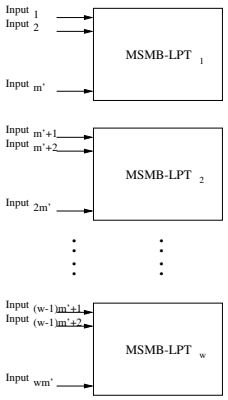

Fig. 2. Block diagram of M-MSMB-LPT scheme.

$1 \le j \le w$, have the same content as $TCAM_u$ in MSMB-LPT of configuration (m', n, k). This scheme is called *multiple MSMB-LPTs* (M-MSMB-LPT for short). We say that an M-MSMB-LPT has configuration (m, n, w, k) if it has w MSMB-LPTs of configuration (m', n, k), where $m = m' \cdot w$. The block diagram of an M-MSMB-LPT scheme is shown in Figure 2.

In an M-MSMB-LPT of configuration (m, n, w, k), w MSMB-LPTs of configuration (m', n, k) operate completely independently. The performance of an M-MSMB-LPT is determined by a single component MSMB-LPT. Each CR can be implemented by a high-performance w-to-1 round-robin arbiter of [15].

B. MSMB-LPT-I

Another design is called *multiple MSMB-LPTs with Interleaved TCAMs* (MSMB-LPT-I for short). An MSMB-LPT-I of configuration (m, n, w, k) has m inputs and wk TCAM blocks that are partitioned into k groups, each called a *TCAM bundle*. Each input is associated with an LPT of size n, and can place a search request to any TCAM bundle through its RD (range detector) in case of an LPT miss. The j-th TCAM bundle, $1 \le j \le k$, has w TCAM blocks, $TCAM_{j,v}$, $1 \le v \le w$. The w TCAM blocks $TCAM_{j,v}$ in the j-th TCAM bundle contain the same content as that of $TCAM_j$ in MSMB-LPT of configuration (m', n, k), where $m' = \frac{m}{w}$. Associated with the j-th TCAM bundle is a CR_j. CR_j (contention resolver) is used to select at most w out of potentially $m = m'w$ search requests according to a priority scheme. Figure 3 is the block diagram of the MSMB-LPT-I scheme.

Let $key_i(n_i)$ denote the n_i-th key from input i, and $result_i(n_i)$ the search result (a prefix) corresponding to $key_i(n_i)$. Initially, $n_i = 0$. Without going into implementation details, the operations of MSMB-LPT, which is a hardware component implemented by an ASIC, are described by a set of concurrent processes *LPT-*$Search_i$, $1 \le i \le m$, and *TCAM-*$Search_j$, $1 \le i \le k$, as follows:.

process *LPT-*$Search_i$
 repeat
  $n_i := n_i + 1$ and get next key $key_i(n_i)$;
  do (a) and (b) **in parallel**

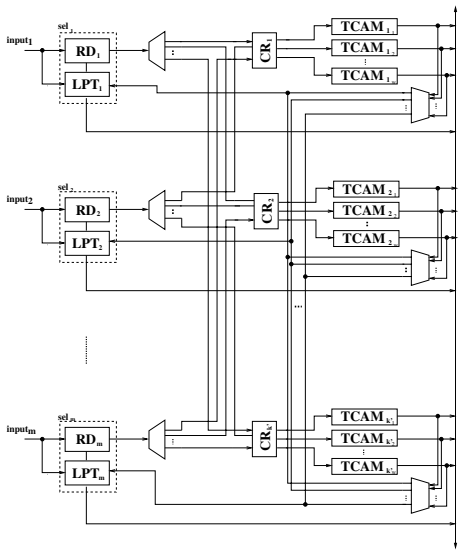

Fig. 3. Block diagram of the MSMB-LPT-I scheme

  (a) RD_i finds j such that key_i is in the j-th TCAM bundle;
  (b) search $key_i(n_i)$ in LPT_i;
 if $result_i(n_i)$ is found in LPT_i **then** output $result_i(n_i)$
 else
  dispatch TCAM search quest $(i, key_i(n_i))$ to CR_j;
  while $HOLD_i$ **do** nothing **end-while**
  use $result_i(n_i)$ to update LPT_i;
 end-repeat
end-*LPT-*$Search_i$

process *TCAM-*$Search_j$
 repeat
  $hold_{p,j} := 1$;
  while there exists idle TCAM blocks and TCAM search requests **do**
   select a waiting request, say (i, key_i);
   find $result_i$ of key_i in an idle TCAM block, say $TCAM_{j,v}$;
   do (a) and (b) **in parallel**
    (a) $hold_{i,j} := 0$; $p := i$;
    (b) output $result_i$
 end-repeat
end-*TCAM-*$Search_j$

Associated with CR_j, there are m signals, $hold_{1,j}$, $hold_{2,j}$, $\cdots$, $hold_{m,j}$. And associated with Sel_i, there is a signal $HOLD_i = hold_{i,1} \wedge hold_{i,2} \wedge \cdots, hold_{i,k}$. Normally, all $hold$ signals are set to 1. Signal $hold_{i,j}$ is set to 0 by some $TCAM$ of the TCAM bundle associated to CR_j has just completed a search on behalf of input i, and it is set back to 1 after the search result is sent to the output. $HOLD_i$ and $hold_{i,j}$ are used as part of synchronization signals.

The concurrent *TCAM-Search* processes for the j-th TCAM bundle are coordinated by CR_j, which can be implemented as a round-robin m-to-w selector introduced in [16].

Clearly, MSMB-LPT-I of configuration $(m, n, 1, k)$ is MSMB-LPT of configuration (m, n, k).

IV. Experimental Results

We conducted a series of simulations on M-MSMB-LPT and MSMB-LPT-I. We tried to make simulations and comparisons as fair as possible. First-in-first-out (FIFO) replacement policy, which is easy to implement in practice, is used for LPT update. Round-robin arbitration, which is easy to be implemented in high-speed as shown in [15], [16], is used for TCAM contention resolution in CRs of M-MSMB-LPT and MSMB-LPT-I. and M-MSMB-LPT-I.

Two packet traces are used in our simulations. Each trace consists of m packet streams, one for each input. Packet trace 1 is generated according to routing table features described in [17] and Internet application traffic characteristics described in [18]. Packet trace 2 is derived from actual packet flows given in [19].

It is important to point out that the performance of some configurations of MSMB-LPT and M-MSMB-LPT can be derived from the performance of MSMB-LPT-I schemes with configurations (m, n, w, k) as follows:

- The performance of MSMB-LPT-I with configuration $(m, n, 1, k)$ is the performance of MSMB-LPT with configuration (m, n, k).
- The performance of MSMB-LPT-I with configuration $(m, n, 1, k)$ is the performance of M-MSMB-LPT with configuration $(w \cdot m, n, w, k)$.

For example, the performance results of MSMB-LPT-I with configuration $(6, n, 1, 4)$ can be used to indicate the performance of M-MSMB-LPT with configuration $(12, n, 2, 4)$ as well as the performance of M-MSMB-LPT with configuration $(18, n, 3, 4)$.

A. Performance metrics

We are interested in the performance of M-MSMB-LPT and MSMB-LPT-I in terms of speedup and power consumption where contention ratio, speedup and power consumption are defined and measured as follows.

Let $C_{MSMB-LPT-I}$ denote the number of contentions at TCAM blocks when an MSMB-LPT-I scheme is applied to a trace. Define the TCAM *contention ratio* of MSMB-LPT-I as

$$CR_{MSMB-LPT-I} = \frac{C_{MSMB-LPT-I}}{N}.$$

We define the MSMB scheme without caching (i.e. without LPTs) as a *naive MSMB* scheme. Let T_{MSMB} be the total key search time of a naive MSMB under a packet trace, and let $T_{MSMB-LPT-I}$ be the total key search time of the corresponding MSMB-LPT-I under the same packet trace. The *speedup* of MSMB-LPT-I over naive MSMB is defined as

$$S_{MSMB-LPT-I} = \frac{T_{MSMB}}{T_{MSMB-LPT-I}}.$$

The power consumption of an M-MSMB-LPT or MSMB-LPT-I M-MSMB-LPT-I scheme is dominated by the power consumptions of its TCAM blocks during its normal operations, since TCAMs are the main source of power consumption in an IP lookup engine. This, combining with the fact that the size of an LPT is much smaller than that of a TCAM block, leads us to measure the power consumptions of M-MSMB-LPT, MSMB-LPT-I and M-MSMB-LPT schemes by the power consumed by their TCAM blocks.

For MSMB-LPT-I with configuration (m, n, w, k), let C be the total number of parallel clock cycles to complete the IP lookup for all packets in a trace, $A_{MSMB-LPT-I_j}$ the total number of clock cycles in which $TCAM_j$ block is searched, and $B_{MSMB-LPT-I} = \sum_1^k A_{MSMB-LPT-I_j}$. Define the *TCAM utilization* of MSMB-PT as

$$U_{MSMB-LPT-I} = \frac{B_{MSMB-LPT-I}}{C \cdot k}.$$

The power consumption of MSMB-LPT-I with configuration (m, n, w, k) is measured by

$$P_{MSMB-LPT-I} = U_{MSMB-LPT-I} \cdot k.$$

In the rests of this section, we report the speedup, contention ratio, and power consumption of M-MSMB-LPT and MSMB-LPT-I schemes. For each performance metric, we conduct the simulation with various configurations under both trace 1 and trace 2. To avoid the redundancy, we report our simulation results under very representative configurations in this paper.

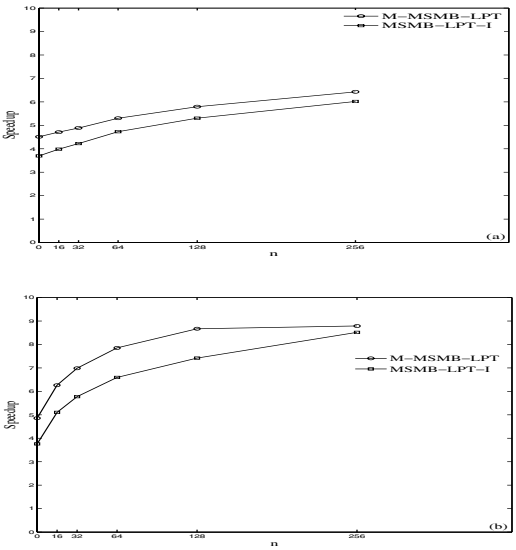

Fig. 4. (a) Speedup of M-MSMB-LPT with configuration $(36, n, 12, 4)$ and MSMB-LPT-I with configuration $(36, n, 4, 4)$ under trace 1; (b) Speedup of M-MSMB-LPT with configuration $(36, n, 12, 4)$ and MSMB-LPT-I with configuration $(36, n, 4, 4)$ under trace 2.

B. Comparison of M-MSMB-LPT and MSMB-LPT-I

Our main focus in simulation is to provide insightful observation on the maximum hardware cost reduction that MSMB-LPT-I can deliver in order to achieve the speedup similar to that of M-MSMB-LPT rather than comparing the speedup of different schemes using the same number of TCAM chips.

978-1-4244-2219-7/08/$25.00 (c)2008 IEEE

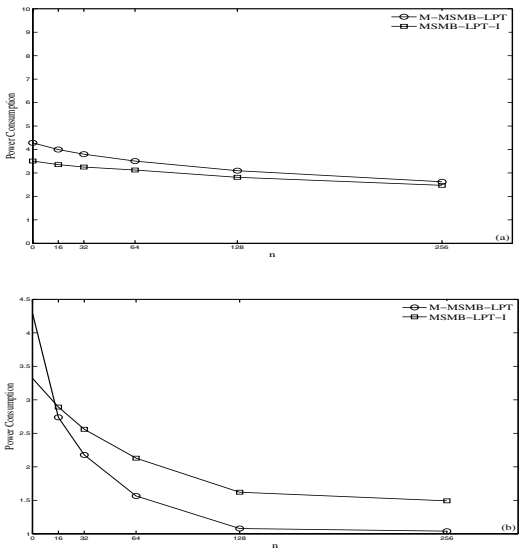

Fig. 5. (a) Power consumption of M-MSMB-LPT with configuration $(36, n, 12, 4)$ and MSMB-LPT-I with configuration $(36, n, 4, 4)$ under trace 1; (b) Power consumption of M-MSMB-LPT with configuration $(36, n, 12, 4)$ and MSMB-LPT-I with configuration $(36, n, 4, 4)$ under trace 2.

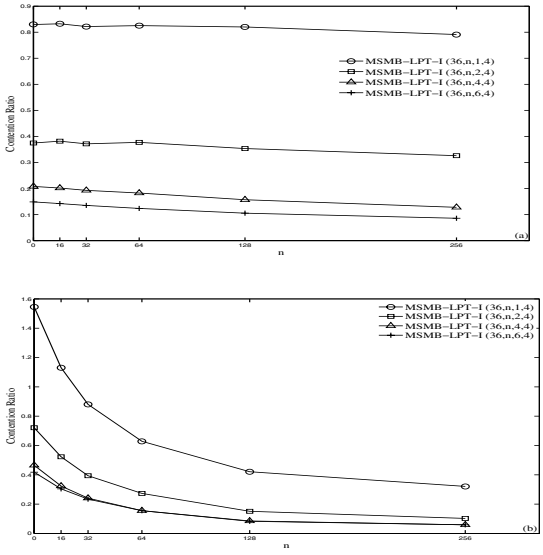

Fig. 6. (a) Contention ratio in MSMB-LPT-I with configuration $(36, n, w, 4)$ under trace 1; (b) Contention ratio in MSMB-LPT-I with configuration $(36, n, w, 4)$ under trace 2

As shown in Figure 4, MSMB-LPT-I with configuration $(36, n, 4, 4)$ can achieve similar speedup as M-MSMB-LPT with configuration $(36, n, 12, 4)$ under both trace 1 and trace 2. There are 48 TCAM chips in M-MSMB-LPT with configuration $(36, n, 12, 4)$, and only 16 TCAM chips in MSMB-LPT-I with configuration $(36, n, 4, 4)$. This demonstrates that significant number of TCAM chips can be reduced using MSMB-LPT-I compared with M-MSMB-LPT schemes in order to achieve similar speedup.

Though the TCAM utilization in MSMB-LPT-I is higher than that in M-MSMB-LPT, fewer number of TCAM chips are used in MSMB-LPT-I. As shown in Figure 5, the power consumption in MSMB-LPT-I with configuration $(36, n, 4, 4)$ is very close to that in M-MSMB-LPT with configuration $(36, n, 12, 4)$.

In Figure 6, we show the case with 36 inputs and 4 TCAM chips in each bundle. When we increase the number of TCAM bundles from 1 to 2, the contention ratio can be dramatically reduced. When we increase the number of TCAM bundles from 4 to 6, not much contention ratio can be further reduced. This simulation result is useful in making a decision on the tradeoff between the TCAM hardware cost and the contention ratio.

Given the available TCAM resource such as the number of TCAM bundles, w, and the number of TCAM chips in each bundle, k, it is important to know the expected contention ratio under different inputs. In Figure 7, there are 2 TCAM bundles and there are 4 TCAM chips in each bundle. When there are only 6 inputs, the contention ratio is very low. The contention ratio increases with the number of inputs increasing. When there are 36 inputs, the contention ratio can be as high as 70% which indicates that more TCAM bundles or more TCAM

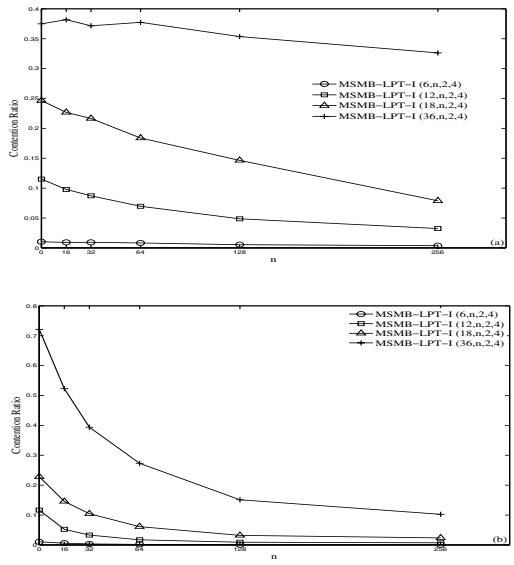

Fig. 7. (a) Contention ratio in MSMB-LPT-I with configuration $(m, n, 2, 4)$ under trace 1; (b) Contention ratio in MSMB-LPT-I with configuration $(m, n, 2, 4)$ under trace 2

chips in each TCAM bundle are needed.

Our next experiment is designed to study the speedup gain of increasing the TCAM bundle for a given number of inputs. In Figure 8, there are 36 inputs. There are 4 TCAM chips in each TCAM bundle. The higher speedup can be achieved when we increase w from 1 to 2, or 4. Very marginal speedup increase can be achieved when we increase w from 4 to 6.

C. Performance study of MSMB-LPT-I

Our last experiment is designed to study how the speedup changes with the number of inputs. It is not a surprise to see

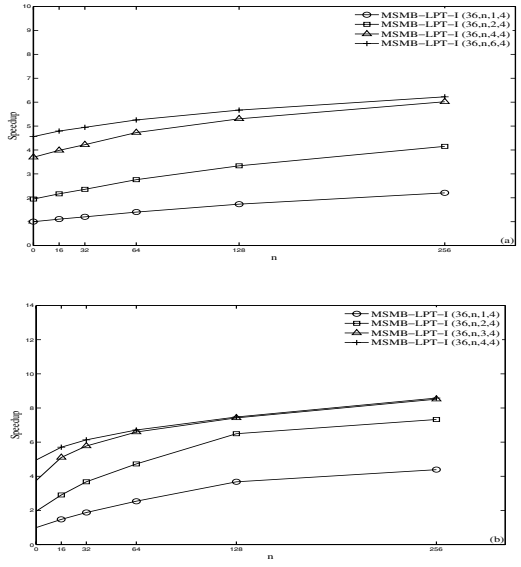

Fig. 8. (a) Speedup of MSMB-LPT-I with configuration $(36, n, w, 4)$ under trace 1; (b) Speedup of MSMB-LPT-I with configuration $(36, n, w, 4)$ under trace 2

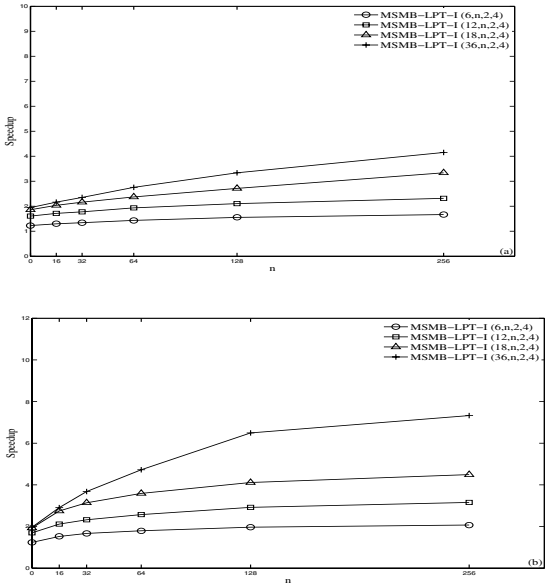

Fig. 9. (a) Speedup of MSMB-LPT-I with configuration $(m, n, 2, 4)$ under trace 1; (b) Speedup of MSMB-LPT-I with configuration $(m, n, 2, 4)$ under trace 2

that the speedup increases with the number of inputs increasing as shown in Figure 9. This result demonstrates the scalability of MSMB-LPT-I. Given the superiority of MSMB-LPT-I, we are interested in studying how contention ratio and speedup respond to the changes of parameters w and m in MSMB-LPT-I. The results are demonstrated in Figures 6 to 9.

V. Concluding Remarks: Tradeoffs and Scalability

We studied two TCAM-based IP lookup schemes, namely M-MSMB-LPT and MSMB-LPT-I. For a given m of inputs, feasible combinations of different n, w and k values give rise to a feasible solution space for different designs of hardware IP lookup engines using these schemes. Tradeoffs among cost, speed, power consumption and modularity (reliability) can be sought. In some situations, one must consider the scalability issue, in which the focus is how to maintain system performance while increasing the input size m. Our hierarchy of MSMB schemes provide a spectrum of possibilities for achieving scalability.

We conducted preliminary simulations. As implementing cache memories in a computer system, before implementing a physical design of TCAM-based IP lookup engine, extensive simulations at different levels should be performed to obtain an optimized design. Our hierarchy of MSMB schemes provide many choices for optimized design.

References

[1] K. Sklower, "A tree-based packet routing table for Berkeley Unix, Technical Report, University of California, Berkeley, 1993.

[2] M. Valdvogel, G. Varghese, J. Turner and B. Plattner, "High speed scalable IP lookups," *Proc. of ACM SIGCOMM '97*, 1997.

[3] S. Nilsson and G. Karlsson, "Fast address lookup for Internet routers," In P. Kuhn and P. Ulrich, editors, *Broadband Communications: The future of Telecommunications*, pp. 11-22, 1998.

[4] V. Srinivasan and G. Varghese, "A survey of recent IP lookup schemes," *Proc. IFIP 6th International Workshop on Protocols for High Speed Networks*, 1999.

[5] A. McAuley and P. Franis, "Fast routing table lookup using CAMs," *Proc. IEEE INFOCOM '93*, Vol. 3, pp. 1382-1391, 1993.

[6] Panigrahy, R. and Sharma, S. "Reducing TCAM power consumption and increasing throughput," *10th IEEE Symposium on High Performance Interconnects Hot Interconnects*(*HOTI'02*), 2002.

[7] Zane, F.,Girija Narlikar, and Basu, A. "CoolCAMs: power-efficient TCAMs for forwarding engines," *Proc. IEEE INFOCOM 2003,* vol. 1, pp. 42-52, 2003.

[8] Kai Zheng, Chengchen Hu, Hongbin Liu, and Bin Liu, "An ultra high throughput and power efficient TCAM-based IP lookup engine," *Proc. IEEE INFOCOM,* vol. 3, pp. 1984-1994, 2004.

[9] Akhbarizadeh, M.J. Nourani, M. Panigrahy, and R. Sharma, "High-speed and low-power network search engine using adaptive block-selection scheme,"

[10] H. Yu, J. Chen, J. Wang, S.Q. Zheng, and M. Nourani, "An improved TCAM-based IP lookup engine," submitted to *HPSR 2008*, 2008

[11] D. Duffy, A. McIntosh, M. Rosenstein, and W. Willinger, "Statistic analysis of CCSN/SSN traffic data from working CCS subnetworks," *IEEE J. Selected Areas in Comm.,* vol. 12, no. 3, pp.544-551, 1994

[12] Matthias Grossglauser and Jean Bolot, "On the relevance of long-range dependence in network traffic," *Proc. ACM SIGCOMM'96,* 1996

[13] Vern Paxson and Sally Floyd, "Wide area traffic: the failure of Poisson modeling," *Proc. ACM SIGCOMM'95,* 1995

[14] Akhbarizadeh and M.J. Nourani, M, "Efficient prefix cache for network processors," *12th IEEE Symposium on High Performance Interconnects* (*HOTI'04*), 2004.

[15] S. Q. Zheng and M. Yang, "Algorithm-hardware codesign of fast parallel round-robin arbiters," *Transactions on Parallel and Distributed Systems*, vol. 18, no. 1, pp. 84-95, 2007.

[16] S. Q. Zheng, M. Yang, and F. Masetti-Placci, "Constructing schedulers for high-speed, high-capacity switches/routers," *International Journal of Computers and Applications*, vol. 25, no. 4, pp. 264-271, 2003.

[17] "BGP routing table analysis report," *http://bgp.potaroo.net*, 2004

[18] "How much information? 2003," *http://www2.sims.berkeley.edu/research/projects*, 2003

[19] "30 days of wide-are TCP connections," *http://ita.ee.lbl.gov/traces.html*, 1993.