IEICE TRANS. COMMUN., VOL.E93–B, NO.11 NOVEMBER 2010

PAPER

A New TCAM Architecture for Managing ACL in Routers

Haesung HWANG[†a)], ***Student Member***, **Shingo ATA**[††], ***Member***, **Koji YAMAMOTO**[†††], ***Nonmember***, **Kazunari INOUE**[†††], ***Member***, ***and*** **Masayuki MURATA**[†], ***Fellow***

SUMMARY Ternary Content Addressable Memory (TCAM) is a special type of memory used in routers to achieve high-speed packet forwarding and classification. Packet forwarding is done by referring to the rules written in the routing table, whereas packet classification is performed by referring to the rules in the Access Control List (ACL). TCAM uses more transistors than Random Access Memory (RAM), resulting in high power consumption and high production cost. Therefore, it is necessary to reduce the entries written in the TCAM to reduce the transistor count. In this paper, we propose a new TCAM architecture by using Range Matching Devices (RMD) integrated within the TCAM's control logic with an optimized prefix expansion algorithm. The proposed method reduces the number of entries required to express ACL rules, especially when specifying port ranges. With less than 10 RMDs, the total number of lines required to write port ranges in the TCAM can be reduced to approximately 50%.

key words: *Access Control List (ACL), hardware cost, IP router, prefix expansion, Ternary Content Addressable Memory (TCAM)*

1. Introduction

The need for high-speed Internet communication has prompted many researchers to design fast network architectures. One of the crucial components in any architecture is the router. Routers are network devices used for forwarding and classifying packets and usually consist of a specialized operating system and memory. More specifically, when a router forwards the packets, it uses forwarding tables to associate destination networks with output ports. Upon every packet arrival, the table is searched for the appropriate entry specifying which port and the IP address the packet should be forwarded to. In the case of access control, the Access Control List (ACL) is searched to determine whether the packet should be permitted or denied, causing the packets to be forwarded to the destination or to be dropped.

According to [1], the rules in an ACL of a typical enterprise network gateway consist of approximately 5,000 entries, a number that is also expected to increase due to greater awareness of network security. Access control policy is becoming more restrictive and shifting from "reject unnecessary traffic" to "transmit only the minimal amount of legitimate traffic" leading to a tendency toward more rules defining which specific packets are legitimate rather than just denying a group of packets. Therefore, in both cases, it is necessary to search for a certain object that quickly matches the given criteria from an enormous candidate set.

These ACLs are written in high-end routers in a type of memory called Content Addressable Memory (CAM). Unlike Random Access Memory (RAM), which uses a memory address as a search key and returns the content of the memory as the result, CAM is a device that searches using the content of the memory and returns the address where the supplied data was found.

There are two kinds of CAM. Binary CAM (BCAM) only returns an entry that exactly matches the input data as it uses `0`s and `1`s, whereas Ternary CAM (TCAM) can also perform partial matches of the entries because it can consist of `0`, `1` or `*`. The "`*`" represents a "don't care" bit in each memory cell. A basic description of TCAM functionality can be found in [2]–[7]. Storage of the IP addresses of ACLs uses this don't care bit to represent a network level address.

Table 1 shows an example of an ACL. ACL entries typically consist of five fields: source and destination IP addresses, source and destination port numbers, and protocol type. Packets are classified according to the corresponding rule and a necessary action (permit/deny) is performed only when all fields are matched. For example in Table 1, access-list 101 permits the TCP packets with source IP address as 10.1.1.2 and with destination address as 172.16.1.1 that equals telnet.

One of the other main advantages of the TCAM besides its search speed is its ability to express ranges. A typical type of data in the form of a range is the port number field of an ACL, which is used in the form of range to allow/deny various applications requiring several port numbers as a set as in video plus audio chat. In BCAM, it is impossible to represent the range without writing every corresponding number within the range in the memory. For example, when writing the range `1024-65535` in a BCAM entry, the simplest form is to write every single number to exactly match the entire entry. However, it is obvious that 64,512 entries are required to write a single range which ends up consuming a huge number of entries in BCAM. We refer to this method as *Full Expansion* in this paper.

Since full expansion is not a practical method, in most cases the range has to be divided into several subranges to

Manuscript received November 26, 2009.
Manuscript revised June 3, 2010.

[†]The authors are with the Graduate School of Information Science and Technology, Osaka University, Suita-shi, 565-0871 Japan.
[††]The author is with the Department of Information and Communication Engineering, Osaka City University, Osaka-shi, 558-8585 Japan.
[†††]The authors are with the Renesas Electronics Corporation, Itami-shi, 664-0005 Japan.
a) E-mail: h-hwang@ist.osaka-u.ac.jp

DOI: 10.1587/transcom.E93.B.3004

Copyright © 2010 The Institute of Electronics, Information and Communication Engineers

Table 1 Example of access control list.

```
access-list 101 permit tcp host 10.1.1.2 host 172.16.1.1 eq telnet
access-list 102 deny tcp any range 137 139 any
access-list 103 permit ip 10.1.1.0 0.0.0.255 172.16.1.0 0.0.0.255
access-list 111 deny icmp any 10.1.1.0 0.0.0.255 echo
access-list 191 permit udp any any range 16384 16483
```

Table 2 Prefix expansion of 1024-65535.

1***************	32768-65535
01**************	16384-32767
001*************	8192-16383
0001************	4096-8191
00001***********	2048-4095
000001**********	1024-2047

be stored in the memory. The process of dividing ranges is called *Prefix Expansion (PE)*. Using TCAM's capability to use "*," PE expresses several port numbers as a single entry by aggregating the least significant bits as "*," which makes it possible to represent subranges in units of 2^i and those aligned on boundaries of powers of 2. The range `1024-65535` would then be represented by the following six lines as in Table 2.

Compared to full expansion, the prefix expansion method can significantly reduce the number of TCAM entries needed. However, configuration of ACL is usually performed manually and ranges based on decimal values (e.g. 10–20, 100–200) are commonly used. This might be a convenient setting for a human operator to understand, but it results in superfluous entries in TCAM. This causes problems when TCAM exhaustion occurs. Partial application of the ACL occurs via software and the packets that match the rules that are not applied in the TCAM are processed by software [8], making wire speed search unachievable.

Despite the advantages of the TCAM, it also has a major drawback. In order to express a don't care bit, TCAM uses 16 transistors per a memory cell, whereas RAM uses 6. Therefore, it is desirable to write data in the TCAM more compactly, i.e. reduce the number of TCAM entries, using less space to save the number of the transistors needed.

Several methods for reducing the number of entries have been suggested. In [9], the *Dynamic Range Encoding Scheme* is suggested. This scheme encodes a subset of the ranges in the ACL and maps it to unused bits in each entry of the TCAM. Each encoded range affects other ranges which further reduces entries; an entry expansion ratio of 1.23 can be achieved, whereas an average full expansion ratio is approximately 6.20 [1]. However, the proposed algorithm uses TCAM resources when encoding the range and ends up accessing TCAM more frequently when there are more ranges to encode. Also, it needs additional memory for mapping, which can result in an increased search time.

In [10], the subranges in the ACL are processed to be in powers of 2 under the premise that the range ends up being semantically unchanged. The total number of required entries decreases by 50%. However, the total reduction rate depends on the inter-dependent ACL rules whereas our proposal focuses on reducing the number of required TCAM entries to store a single port range which is independent of the ranges. Also, [10] approaches the given issue with a software method (trimming, expanding, adding, and merging) whereas our proposal uses a hardware method (NOT/AND/OR gates) to achieve a faster processing speed.

In [11], the ranges are encoded using ternary values, but the major difference from the two papers mentioned above is that it places don't care bits at arbitrary places instead of in a prefix form. The result also uses unused bits in each entry. Using 32 bits for this encoding, the suggested algorithm can reduce the necessary entries by up to 50%. However, the author mentions that the optimal encoding method depends on the ACL's content which cannot be known *a priori*.

Cisco's IP routers use a device called Layer-4 Operation Unit (L4Op) to write port numbers in ranges in order to make use of the limited TCAM resources [12]. L4Op is a hardware device which resides outside of the TCAM and makes it possible for the ranges to be logically compared using the operators *gt (greater than)*, *lt (less than)*, *range*, and *neq (not equal)*, and determines if the input port number matches the specified condition by matching every other field except for the port ranges. Only after a positive result does it begin to determine whether logical terms of port ranges also match. By this method, port fields in ranges are only managed under L4Op which is independent of the TCAM itself. As a result, the possibility of expressing a rule in a single TCAM entry has the effect of writing more rules to TCAM in the end.

However, the biggest problem with this technique is that L4Op is implemented outside of the TCAM circuit. Because the input data format for the L4Op is entirely different from TCAM's, the Input/Output (I/O) band consumption ends up being twice as much, trying to represent the same rate when there is a search from the outside to TCAM. In other words, double the amount of I/O pin numbers is needed under the same frequency. When the physical size of the device is the same, increasing the I/O pin numbers for external I/O causes the distance between the pins to get smaller and ends up requiring higher-precision wire connections. Also, the signal integrity and the SSO (Simultaneous Signal Output) noise become important factors in the recent high speed hardware devices, which creates restrictions in designing boards in order to reduce the interference noise between the pins. Because of this, it is desirable to have a smaller number of pins if the hardware chip performance should remain the same.

These algorithms consist only of software solutions or the implementation of new devices outside of the TCAM

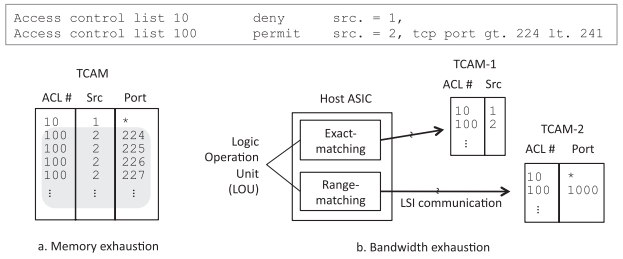


Fig. 1 Two reasons for TCAM exhaustion.

and their actual application is limited. Adding an extension to the TCAM itself and thoroughly considering how the number of entries can be reduced is, to our knowledge, unprecedented work.

Another commercial product, NetLogic's Range Encoding Engine (REE) [13], implements range encoding that allows customers to effectively double the efficiency of performing port range inspection. However, it is very difficult to compare our work with NetLogic's REE because (i) no public document is available, therefore it is difficult to find out about the specifications and (ii) it is unclear if "double the efficiency" means whether the search time halved or the required memory space was saved 50%.

Figure 1 shows two reasons for TCAM exhaustion. Figure 1(a) is when a port range from 224 to 241 ends up occupying memory space and Fig. 1(b) is when devices such as L4Op exhausts bandwidth by handling exact and range matching process separately, therefore increasing I/O pin numbers.

Our proposal regarding the ACL is to use a modified TCAM chip with *Range Matching Devices (RMD)* in order to restrain the growth of TCAM entries through prefix expansion. RMDs are integrated within the TCAM's logic, therefore reducing the cost of extra I/O lines. In addition, it is possible to maintain the conventional TCAM from the user's point of view and simultaneously permit arbitrary ranges to be stored. This reduces hardware costs. In addition, in a situation where the limited number of RMDs is exhausted, we consider the effect of adding logical NOT and AND gates to the TCAM in order to achieve a flexible combination of range expressions. Using this method, we show that it is possible to improve ACLs management compared to the current practice of only using logical OR operations.

In Sect. 2, we describe the proposed RMD logic structure, as well as the TCAM architecture using AND/NOT gates. Also, we proposed a policy for how and what to write in RMDs and explain the prefix expansion algorithm using AND/NOT/OR gates. Section 3 provides an evaluation of our method using the database of an existing network. Finally, in Sect. 4, we conclude the paper and give an outlook on future work.

2. Management of Range Matching Device

In this section we describe the logical circuit diagram of the RMD, propose a policy for storing ranges in it, and clarify the three PE algorithms used in our work. We first describe the mechanism introduced in [14] and evaluate the proposal with updated ACL data.

2.1 TCAM Hardware Implementing Range Specifications

As previously mentioned in the Introduction, the number of entries to be stored in TCAM does not decrease much by applying prefix expansion and it is difficult to optimize and update L4OP. Numerous studies so far have used an existing TCAM device and implemented additional software or external devices to minimize the number of required entries. We aim to support port numbers in ranges by extending the TCAM device.

As an extension to the conventional TCAM, we analyze the effect of adding a range determining circuit and logical operators between the entries using the following methods.

2.1.1 Range Matching Device

Figure 2 is a logic circuit diagram of the proposed RMD. It contains *Normal*, *RMD*, *Src/Dst*, *From*, *To*, *Load Enable* and *Search Line* as inputs and *Out* as an output. *Normal* and *RMD* are used when an user decides whether to use this device in TCAM as a memory cell that has a data or as a pointer to the RMD. To make an RMD contain a range, there has to be a positive signal to *Load Enable* and *Src/Dst* which determines if this is set for the source or destination port number. At the same time, each *From* and *To* line gets an input of 16 bits that represents the range. We briefly explain the symbols used in Fig. 2. Multiplexer selects one of many analog or digital input signals and forwards the selected input into a single line. Flip-flops in the figure work as primitive memory cells, as the output (Q) takes the value of the input (D) and delays it by maximum one clock count. Magnitude comparator is self-explanatory: it compares the two input A and B and returns 1 or 0 depending on the condition.

When the RMD is operated as a searching device, either the source or destination port is selected based on the input information above and is compared with the stored range. Only when the input is between *From* and *To* does it return the result of *Out* as 1 (otherwise 0). In TCAM, separate bits for the RMD have to be assigned in addition to the other data bits such as source/destination IP address, source/destination port, and the protocol. A single bit is assigned for each RMD. The bit, corresponding to the RMD that contains the desired range, is set to `1` and rest of the bits for other RMDs are set to `*`. Figure 3 shows the interconnection of Figs. 2 and 4 with example ranges in RMDs. Let us assume that we have four kinds of ranges that are already stored in the RMD (`6970-6999`, `2326-2837`, `3230-3253`, and `1024-65535`). According to the above RMD bit assigning rule, the RMD part within the TCAM becomes `*1**`. When there is a search key port number 2436, it returns the output of `0100` (Internal Search Key) and entry #3's result is 1, meaning a match. This packet will be either denied

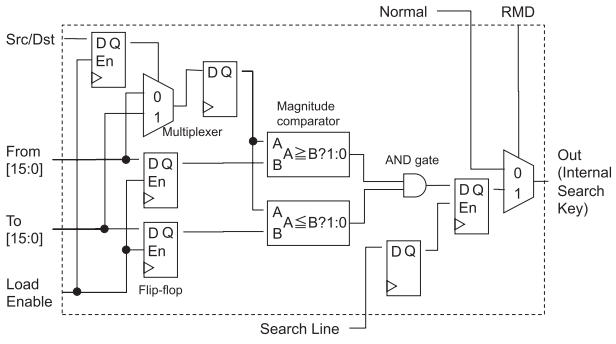


Fig. 2 Example of range matching device.

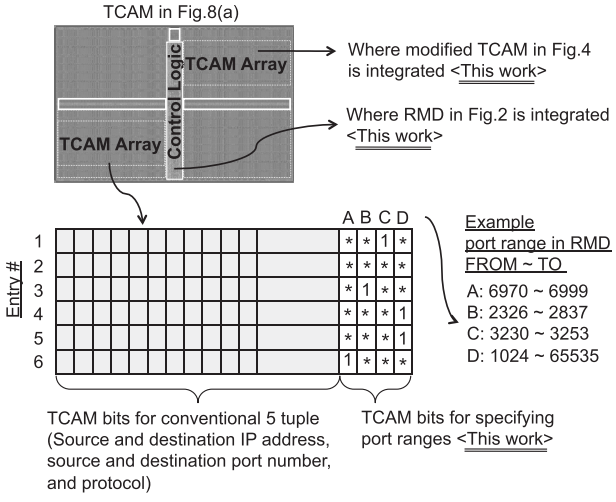


Fig. 3 Bits for range matching device in TCAM and interconnection of Figs. 2 and 4.

or permitted, depending on the specified action in the corresponding ACL rule.

2.1.2 Adding Logical Operational Gates between the Entries

Since TCAM is considered to have performed a successful search when it returns at least one entry that satisfies every field, we can consider a situation in which each entry's result performs a logical OR action with the other entries' results (which we address as PE-OR in this paper). In this paper, in addition to the conventional PE-OR method, we analyze the performance of extending the TCAM capability by adding NOT and AND gates (which we address as PE-MIN in this paper). This is because additional gates such as NOT and AND are necessary to fully express PE-MIN. The algorithm of PE-MIN and an example range decomposition is given in Sect. 2.3. Figure 4 shows a modified TCAM circuit which has NOT/AND gates. NOT gates are first implemented in the left-most part, followed by AND gates. Since it is impossible to perform a logical AND operation in the hardware only when it is needed, we need sets of ANDs (2, 4, 8 in this case), to be already embedded in the hardware. Depending on how many entries we need to represent each

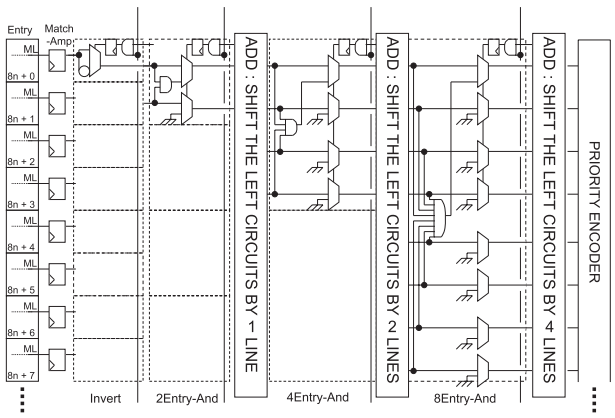


Fig. 4 Additional NOT/AND operation gates in TCAM array.

subrange that uses AND sets, we can pick the smallest AND set needed and write any rules exceeding 8 lines in a regular PE-OR form. The benefit of these extra gates is being able to utilize the PE-MIN, which reduces the total number of entries needed to represent ranges.

The cost estimation of adding NOT and AND gate itself was omitted in Sect. 3 since the manufacturing cost is trivial. For example, from the Fig. 4 for 8 entries, approximately 380 transistors are used; 7 AND gates (2 transistors each $\times$ 7), 23 flip-flops (2 transistors each $\times$ 23), and 32 multiplexers (10 transistors each $\times$ 32). Since a single TCAM bit uses 16 transistors, 9,216 transistors are used in an entry (assuming an entry is 576 bits). Therefore, for 8 entries, transistors in Fig. 4 is less than 0.5% of the transistors used for the actual TCAM.

2.2 RMD Policy

RMDs are limited resources and should be used carefully since they can determine the TCAM's performance specification. Writing ranges that are unpopular or ranges that do not consume multiple numbers of entries after the prefix expansion in to RMDs would not decrease the total number of required entries to the maximum degree. Therefore, in order to minimize the TCAM entries, it is desirable to give a higher priority to the range that has the highest entry reduction ratio when being written to the RMD. In this paper, the weight of each range, which determines the rank to be written in the RMD, is calculated as *(Lines after PE-1)* multiplied by *(Number of ACLs referring this range)*. In other words, a range has a higher tendency to be written into the RMD when it expands to more lines, or has a higher appearance in the ACL database, than other ranges.

2.3 Optimization of the Prefix Expansion

In this section, we present an optimized prefix expansion scheme using NOT/AND/OR operations between entries in the TCAM with the proposed hardware structure in Sect. 2.1.2. Here we briefly explain three prefix expansion algorithms: PE-OR, PE-Exclusive, and PE-MIN. Figure 5

shows a simple representation of the three algorithms with the example range of 5000-6000.

Before demonstrating the algorithms, we first state the condition of a range *From* to *To* that can be expressed with a single TCAM entry. We address this as condition 'single TCAM entry (STE)'.

From: $F = \sum_{i=0}^{15} d_i 2^i$, where $d_i = 0$ or 1

To: $F + 2^j - 1$, where $j = i$ or $i - 1$ and i is $min(i)$ of $d_i = 1$

PE-OR is the conventional prefix expansion method that represents the range in units of powers of 2. It shows especially outstanding performance when the range satisfies the condition STE.

PE-Exclusive first finds the minimum range of STE that entirely covers the given range (4096-6143 in Fig. 5), and expresses the unnecessary parts in combinations of powers of 2. This method is convenient in expressing ranges that are not suitable for PE-OR.

PE-MIN is the most optimized entry reduction algorithm of these three. It uses brute force to find the appropriate ranges. Below is the description of the algorithm.

1. First, do a longest prefix match of $From$ and To in the range $[From, To]$. Select *Reference Point b* by setting the most significant bit position that differs for the first time. Furthermore, let us divide the range into two parts $[From, b]$ and $[b, To]$
2. Let D be the interval from the b and To. For the interval $[b, To]$, find the largest value of i which satisfies $2^i < D \leq 2^{i+1}$
3. For the two ranges of $[b, To]$ and $[b, b + 2^{i+1}]$, calculate x_l and x_r which are the entries needed for each range, respectively
4. If $x_l < x_r$, add a prefix of $[b, To]$, or else add $[b, b+2^{i+1}]$
5. For the range $[To, b + 2^{i+1}]$ which exceeds the original given range, repeat steps 2–4 and apply NOT operation
6. Repeat steps 2–4 for the range $[From, b]$

As shown in Figure 5, the range 5000-6000 in PE-MIN is expressed as ((4992-5119) OR (5120-6143)) AND (NOT (4992-4999) AND (NOT (6000-6015)) AND (NOT (6016-6143))) OR (6000), where b is 5120 (binary: 1010000000000) because 5000 is 1001110001000 and 6000 is 1011101110000 in binary, respectively.

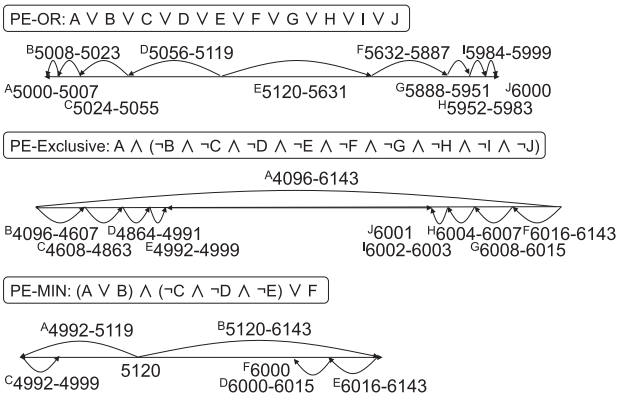


Fig. 5 Prefix expansion algorithms for the range of 5000-6000.

3. Effectiveness of RMD and Prefix Expansion

We now analyze how many entries are required in writing the port numbers expressed in ranges and discuss the result using a real-life database. Table 3 shows the information about the three campus-level ACL sets used in this analysis, which were obtained in April and October 2007, and April 2008.

3.1 Entry Reduction with RMDs

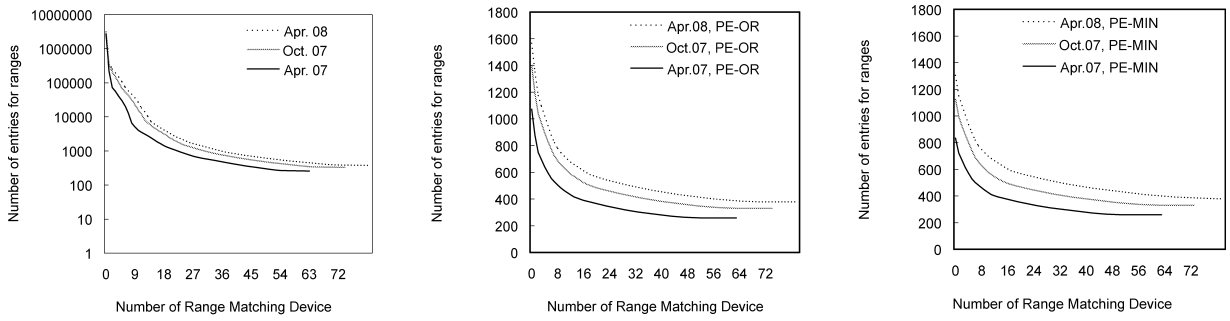
Figure 6 shows how the required entries can be reduced when RMDs are used. Figure 6(a) shows how writing every single number (Full Expansion) that is represented in ranges can consume an enormous amount of TCAM resources. From this figure, we can easily see that we need an order of a million when there are no RMDs. The reason for this phenomenon is that the range 1024-65535 which consisting of registered/dynamic ports is common and it expands to 64,512 lines. By only using one RMD, this range gets written in the memory with the highest priority which results in one tenth of the total required number of lines.

Next, Fig. 6(b) shows the effect of entry reduction when RMDs are used with prefix expansion. We calculate the weight proposed in Sect. 2.2 and write ranges with larger weight first to the RMD. As a result, we can see that it is possible to drastically reduce the number of entries by putting them into RMDs. When the limited number of RMDs is exhausted, ranges with lower priority are written in the TCAM by using prefix expansion. The reduction rate is especially high when a small number of RMDs is implemented and when PE-OR is used: when there are seven to eight RMDs, the total required ranges to write can be reduced as much as 50% from that of not using any RMDs at all. In addition, five to six RMDs can achieve the same effect as in the PE-MIN case (see Fig. 6(c)). Also, all three figures in Fig. 6 show an exponential decrease and after a certain number of RMDs, it is useless to add further RMDs. This is due to the calculation of the weight of each range. Ranges with low priority (i.e., low weight) do not reduce the total number of entries needed even if they are written in the RMD.

Furthermore, when we compare the PE-OR and PE-MIN schemes, the required total entries decrease by about 12% by adding AND/NOT logical gates to the TCAM (614 lines for PE-OR and 695 lines for PE-MIN, both with three RMDs). When PE-OR tries to accomplish the same level as

Table 3 Three ACL sets for evaluation.

Date Captured	Apr. 07		Oct. 07		Apr. 08	
# of unique ACL entries	6,440		7,202		7,703	
	Src	Dest	Src	Dest	Src	Dest
# of ranges	3	256	6	325	6	373
# of unique ranges	63		74		82	



(a) Full Expansion with RMD (b) PE-OR with RMD (c) PE-MIN with RMD

Fig. 6 Comparisons of total required entries (w/ or w/o PE combined with RMD).

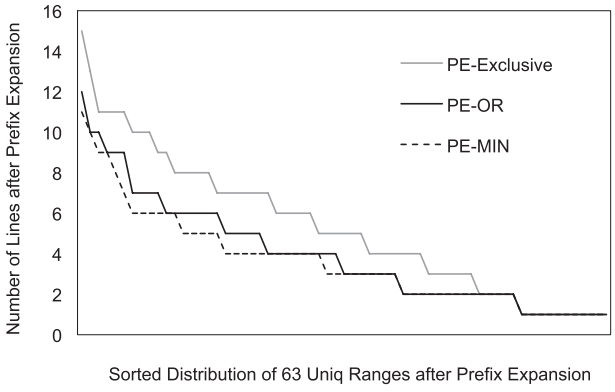
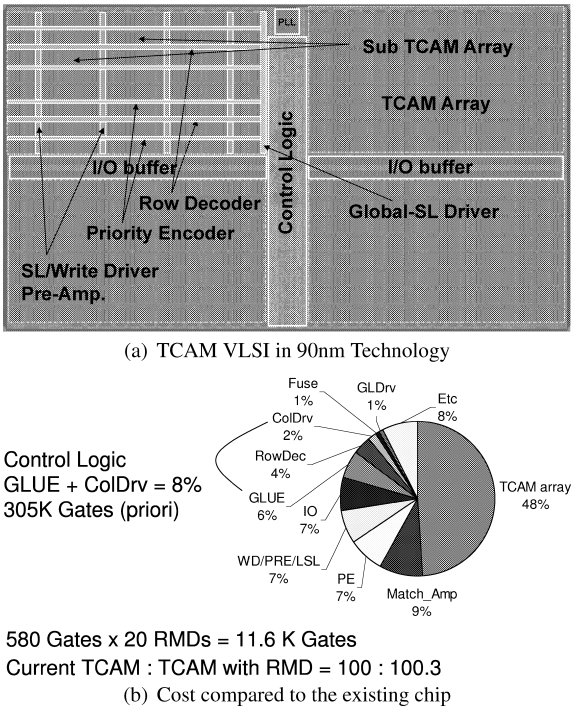


Fig. 7 Comparisons of prefix expansion algorithms.



(a) TCAM VLSI in 90nm Technology

(b) Cost compared to the existing chip

Fig. 8 Existing TCAM and cost comparison with proposed hardware.

that of PE-MIN, two to three further RMDs are needed.

An algorithmic analysis of the different prefix expansions shows the distribution of expanded lines in each method using the April 2007 database. Figure 7 shows how each algorithm expands the range. The total number of lines can be reduced to approximately 10% using the PE-MIN compared to PE-OR. Comparing the entry reduction itself, PE-MIN uses the least TCAM resources, but it requires built-in logical NOT and AND gates in the TCAM itself.

Then, the question arises: which is a better choice? From the point of view only of the number of required TCAM entries, PE-MIN is the definite choice since it can minimize the number of vertical lines. However, adding NOT/AND gates to fully support PE-MIN might end up increasing the horizontal bits (bits added for RMD, x in Fig. 9(a)) in the TCAM, which is not a highly desirable choice since determining the optimum AND sets might be different for each ACL. PE-OR might seem like an appealing choice since it does not require any additional logical operation gates. The problem with using only PE-OR, however, is that this method is not well-suited for some intentional worst-case tests, like setting worst-case ranges mentioned in Appendix in each source and destination port numbers to create $29 \times 29 = 841$ lines for a single rule. Unless this is the case, using PE-OR might be adequate.

One of the biggest advantages of adding logical NOT/AND gates in addition to minimizing the total entries required in storing the ACL is that it makes it possible for the "except" rules of the ACL to be stored directly in the TCAM, whereas the conventional methods such as Cisco IOS or Juniper Junos prefer to use the software solution.

3.2 Overhead Cost Estimation of Adding RMDs

Finally, we analyze the increase in the hardware cost caused by adding RMDs. Fig. 8(a) shows the TCAM hardware structure using 90 nm technology. The ratio of the components in this chip are shown in Fig. 8(b). Among all of the components, 8% is *Control Logic* which is known to be approximately 305 K gates. Meanwhile, the gate, which is used as a unit for chip area, of a single RMD is 580 gates. When inserting 20 RMDs in TCAM, this results in around 11.6 K gates, and corresponds to 0.3% of the total TCAM

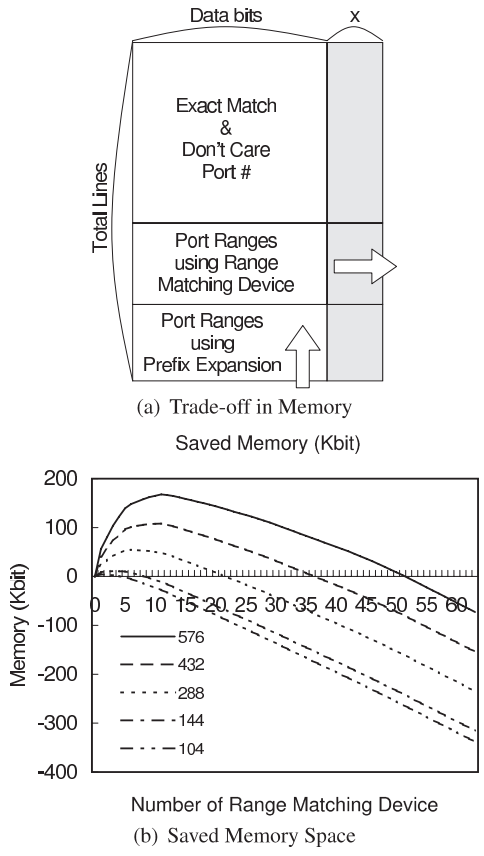


Fig. 9 Wasted and saved memory when range matching devices are used.

gates. Therefore, with the 0.3% increase in manufacturing cost, it is possible to significantly reduce the required number of TCAM lines to write port numbers, which results in more ACL storage space for the limited TCAM resource. We also point out that the component ratio would stay about roughly the same even if the chip technology advances to 65 nm.

Figure 9(a) shows that one additional RMD requires one more TCAM bit to represent the result of the RMD part. Thus RMD reduces the number of vertical lines in TCAM, but at the cost of increasing the horizontal lines. Therefore, it is necessary to consider the trade-off of both factors in order to decide the optimal number of RMDs for implementation in a TCAM device.

Figure 9(b) shows the total saved memory as RMDs are added, according to the calculation in [14]. Five kinds of data bits are considered: 104, 144, 288, 432, and 576 (104 bits for the five-tuple and others for a multiplication of 72 which is a common unit of TCAM cells in current technology). This result shows that memory space is saved in the beginning when the RMDs are first added but after a certain number of RMDs, the overhead increases and it is no longer possible to gain any further benefits. Also, when the data bits are shorter, the wasted memory increases faster as the RMDs are added. The ratio of port numbers expressed in ranges in an ACL affects the optimal number of RMDs required in order to make the most of its advantages.

In the above, we calculated the disadvantage of the increased chip area in gates when 20 RMDs are added. Using Table 3 and Fig. 9, we can find out the trade-off between the added RMDs and reduced memory size in the same unit. Among 6,440 ACL entries (April'07 data), 259 entries use port numbers in range which means the remaining 6,181 entries will consume a single entry. When 20 RMDs are added, this horizontal bits will increase (left to right arrow in Fig. 9(a)); $6{,}181 \times 20 = 123{,}620$ bits. However, using 20 RMDs will reduce the required number of entries (vertical bits, bottom to up arrow in Fig. 9(a)), from 834 to 352, resulting in saved memory of $(832\text{-}352) \times 576 = 277{,}632$ bits (assuming 576 bits are used for a TCAM entry). Therefore, it is approximately 56% beneficial to use 20 RMDs in this case.

The effect of adding RMDs is expected to be bigger in the future, considering that the proportion of ACL will only increase due to the growth in the awareness of network security.

4. Conclusion and Future Work

For the next generation of TCAM in routers, it will be important to design hardware based on the data that is written in TCAM such as ACLs. The benefit is lower power consumption and better utilization of the hardware resources.

We proposed a new TCAM architecture that combines optimized prefix expansion and range matching devices to reduce the number of lines required in TCAM to represent port numbers in ranges. By using real-life ACL databases, we have also shown how the proposed architecture can manage the ACL more effectively than conventional methods, reducing the required number of TCAM entries to express the port numbers in ranges by 50%. The significance of reduced entries is large. This 50% reduction means that the required transistors is halved. Overall power consumption is greatly affected by the number of the transistors [15] therefore, we can claim that the power consumption of the memory array is approximately halved as also. In addition, less space per a chip means more number of chips in a wafer, lowering the production cost.

As future work, analysis of other ACLs using the proposed method to achieve a general-purpose TCAM architecture is needed. Also, we intend to implement the new TCAM in a network processor to conduct further studies on performance characteristics such as power consumption in watt. In addition, the management issues such as addition/deletion of rules and the change in the rule ranks have to be considered.

Acknowledgment

We thank Go Hasegawa and Kenji Leibnitz of Osaka University for their valuable advice and comments.

References

[1] D.E. Taylor and J.S. Turner, "ClassBench: A packet classification benchmark," Technical Report (WUCSE2004-28), Department of Computer Science & Engineering, Washington University, April 2004.
[2] A.J. McAuley and P. Francis, "Fast routing table lookup using CAMs," Proc. IEEE INFOCOM, pp.1382–1391, March 1993.
[3] H. Liu, "Efficient mapping of range classifier into ternary CAM," Proc. Hot Interconnects (HOTI), pp.95–100, Aug. 2002.
[4] F. Zane, G. Nalikar, and A. Basu, "CoolCAMs: Power-efficient TCAMs for forwarding engines," Proc. IEEE INFOCOM, pp.42–52, April 2003.
[5] S. Ahmad and R.N. Mahapatra, "An efficient approach to on-chip logic minimization," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol.15, no.9, pp.1040–1050, 2007.
[6] S. Sharma and R. Panigrahy, "Sorting and searching using ternary CAMs," Proc. Hot Interconnects (HOTI), pp.101–106, Aug. 2002.
[7] B. Agrawal and T. Sherwood, "Modeling TCAM power for next generation network devices," Proc. IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), pp.120–129, March 2006.
[8] Cisco, "ACL and QoS TCAM exhaustion avoidance on catalyst 4500 switches." available at http://www.cisco.com/en/US/products/hw/switches/ps663/products_tech_note0 9186a008054a499.shtml, 2005.
[9] H. Che, Z. Wang, K. Zheng, and B. Liu, "DRES: Dynamic range encoding scheme for TCAM coprocessors," IEEE Trans. Comput., vol.57, no.7, pp.902–915, 2008.
[10] Q. Dong, S. Banerjee, J. Wang, D. Agrawal, and A. Shukla, "Packet classifiers in ternary CAMs can be smaller," Proc. ACM SIGMETRICS/Performance, pp.311–322, June 2006.
[11] K. Lakshminarayanan, Rangarajan, and S. Venkatachary, "Algorithms for advanced packet classification with ternary CAMs," Proc. ACM SIGCOMM, pp.193–204, Aug. 2005.
[12] M.A. Ross, S.D. Chen, and A.V. Bechtolsheim, "Logical operation unit for packet processing," US patent 6,658,002, Dec. 2003.
[13] NetLogic, "Range encoding engine (REE)." available at `http://www.netlogicmicro.com/4-news/pr/2008/08-07-21.htm`
[14] H. Hwang, K. Yamamoto, S. Ata, K. Inoue, and M. Murata, "Minimization of ACL storage by adding minimal hardware of range matching and logical gates to TCAM," Proc. International Conference on High Performance Switching and Routing (HPSR), pp.116–122, May 2008.
[15] N. Wehn and M. Muench, "Minimizing power consumption in digital circuits and systems — An overview," Kleinheubacher Berichte, vol.43, pp.308–319, 2000.

Appendix: Example of Prefix Expansion (worst-case)

In Figure A·1, we demonstrate the worst case of PE-OR, range `16385-65534`, along with PE-Exclusive and PE-Min.

29 entries are needed to express the range `16385-65534` with the conventional method in TCAMs. In another words, in order to determine if a packet specifying port number `20000` should be accepted or denied, it has to match at least an entry from the 29 TCAM lines. However, this 29 entries can be significantly reduced by using PE-Exclusive or PE-Min. The logical expressions are,

PE-Exclusive:

```
(0-65535) AND (NOT 0-16383) AND
(NOT 16384) AND (NOT 65535)
```

1) PE-OR for range 16385 ~ 65534

1 :	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	16385
2 :	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	*	16386 ~ 16387
3 :	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	*	*	16388 ~ 16391
4 :	0	1	0	0	0	0	0	0	0	0	0	0	0	1	*	*	*	16392 ~ 16399
5 :	0	1	0	0	0	0	0	0	0	0	0	0	1	*	*	*	*	16400 ~ 16415
6 :	0	1	0	0	0	0	0	0	0	0	0	1	*	*	*	*	*	16416 ~ 16447
7 :	0	1	0	0	0	0	0	0	0	0	1	*	*	*	*	*	*	16448 ~ 16511
8 :	0	1	0	0	0	0	0	0	0	1	*	*	*	*	*	*	*	16512 ~ 16639
9 :	0	1	0	0	0	0	0	0	1	*	*	*	*	*	*	*	*	16640 ~ 16895
10 :	0	1	0	0	0	0	0	1	*	*	*	*	*	*	*	*	*	16896 ~ 17407
11 :	0	1	0	0	0	0	1	*	*	*	*	*	*	*	*	*	*	17408 ~ 18431
12 :	0	1	0	0	0	1	*	*	*	*	*	*	*	*	*	*	*	18432 ~ 20479
13 :	0	1	0	0	1	*	*	*	*	*	*	*	*	*	*	*	*	20480 ~ 24575
14 :	0	1	0	1	*	*	*	*	*	*	*	*	*	*	*	*	*	24576 ~ 32767
15 :	1	0	0	*	*	*	*	*	*	*	*	*	*	*	*	*	*	32768 ~ 49151
16 :	1	1	0	0	*	*	*	*	*	*	*	*	*	*	*	*	*	49152 ~ 57343
17 :	1	1	1	0	0	*	*	*	*	*	*	*	*	*	*	*	*	57344 ~ 61439
18 :	1	1	1	1	0	0	*	*	*	*	*	*	*	*	*	*	*	61440 ~ 63487
19 :	1	1	1	1	1	0	0	*	*	*	*	*	*	*	*	*	*	63488 ~ 64511
20 :	1	1	1	1	1	1	0	0	*	*	*	*	*	*	*	*	*	64512 ~ 65023
21 :	1	1	1	1	1	1	1	0	0	*	*	*	*	*	*	*	*	65024 ~ 65279
22 :	1	1	1	1	1	1	1	1	0	0	*	*	*	*	*	*	*	65280 ~ 65407
23 :	1	1	1	1	1	1	1	1	1	0	0	*	*	*	*	*	*	65408 ~ 65471
24 :	1	1	1	1	1	1	1	1	1	1	0	0	*	*	*	*	*	65472 ~ 65503
25 :	1	1	1	1	1	1	1	1	1	1	1	0	0	*	*	*	*	65504 ~ 65519
26 :	1	1	1	1	1	1	1	1	1	1	1	1	0	0	*	*	*	65520 ~ 65527
27 :	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	*	*	65528 ~ 65531
28 :	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	*	65532 ~ 65533
29 :	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	65534

2) PE-Exclusive for range 16385 ~ 65534

1 :	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	0 ~ 65535
2 :	0	0	*	*	*	*	*	*	*	*	*	*	*	*	*	*	0 ~ 16383
3 :	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16384
4 :	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	65535

3) PE-Min for range 16385 ~ 65534

1 :	0	1	*	*	*	*	*	*	*	*	*	*	*	*	*	*	16384 ~ 32767
2 :	1	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	32768 ~ 65535
3 :	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16384
4 :	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	65535

Fig. A·1 Prefix expansion algorithms for range of `16358-65534`.

PE-Min:

```
((16384-32767) OR (32768-65535)) AND
(NOT 16384) AND (NOT 65535).
```

Here, both PE-Exclusive and PE-Min require four entries to express the given range but in most cases, PE-Min has the least number of required entries.

Haesung Hwang received her B.E. degree in School of Electrical Engineering and Computer Science from Kyungpook National University, Korea and M.E. degree in the Graduate School of Information Science and Technology, Osaka University in 2005 and 2009, respectively. Currently she is a Ph.D. candidate. Her research interests include design and evaluation of Ternary CAM, access control lists, and content based routing. She is a student member of IEEE.

Shingo Ata received M.E. and D.E. degrees in Informatics and Mathematical Science from Osaka University in 1998 and 2000, respectively. Since 2006, he has been an Associate Professor of Graduate School of Engineering at Osaka City University. His research work includes networking architecture, design of communication protocols, and performance modeling of communication networks. He is a member of IEEE and ACM.

Koji Yamamoto received the A.A.S. degree in Electrical Engineering from Tokushima University, Japan in 1989. In 1989, he joined the Mitsubishi Electric Engineering Co. Ltd. Hyogo, Japan, where he had been engaged in the design and development of Application Specific Memory. In 2003, he moved to the Renesas Design Corporation. He is currently an engineer of Ternary CAM.

Kazunari Inoue received his B.E. and D.E. degrees in Information Technology from Yokohama National University and Research Institute for Nano-device and Systems from Hiroshima University, Japan in 1984 and 2005, respectively. In 1984, he joined Mitsubishi Electric Corporation and moved to Renesas Technology Corporation in 2003, where he has been engaged in the development of LSI and solutions in network applications, research and design of switching, routing and buffering. He is a member of IEICE and Semiconductor Technology Academic Research Center (STARC).

Masayuki Murata received his M.E. and D.E. degrees from Osaka University, Japan, in 1984 and 1988 respectively. In 1984, he joined the Tokyo Research Laboratory, IBM Japan, as a researcher. He was an assistant professor from 1987 and an associate professor at Osaka University from 1992 to 1999. Since 1999, he has been a professor at Osaka University, and he is now with Graduate School of Information Science and Technology, Osaka University. He has more than 500 papers in international and domestic journals and conferences. His research interests include computer communication networks, performance modeling and evaluation. He is a member of IEEE and ACM.