Worst-Case TCAM Rule Expansion

Ori Rottenstreich and Isaac Keslassy
Department of Electrical Engineering
Technion - Israel Institute of Technology
Haifa 32000, Israel
{or@tx,isaac@ee}.technion.ac.il

***Abstract*—Designers of TCAMs (Ternary CAMs) for packet classification deal with unpredictable sets of rules, resulting in highly variable rule expansions, and rely on heuristic encoding algorithms with no reasonable expansion guarantees. In this paper, given several types of rules, we provide new upper bounds on the TCAM worst-case rule expansions. In particular, we prove that a W-bit range can be encoded using W TCAM entries, improving upon the previously-known bound of $2W-5$. We also propose a modified TCAM architecture that uses additional logic to significantly reduce the rule expansions, both in the worst case and in experiments with real-life classification databases.**

I. Introduction

A. Background

Packet classification is the core function behind many network applications, such as routing, filtering, intrusion detection, accounting, monitoring, load-balancing, policy enforcement, differentiated services, virtual routers, and virtual private networks [1]–[4]. For each incoming packet, a packet classifier compares the packet header fields against a list of rules, e.g. taken from access control lists (ACLs). Then, it returns the first rule that matches the header fields, and applies a corresponding action on the packet.

Today, hardware-based TCAMs (Ternary Content-Addressable Memories) are the standard devices for high-speed packet classification [5], [6]. TCAMs are associative-memory devices that match packet headers using fixed-width ternary arrays composed of 0s, 1s, and $*$s (don't-care). For each packet, TCAM devices can check all rules in parallel, and therefore can typically reach higher line rates than software-based classification algorithms [1]–[3]. For instance, the 55 nm CMOS-based NLA9000XT TCAM device can run nearly 500 million searches per second [5].

However, *power consumption* constitutes a bottleneck for TCAM scaling [7]. Given the same access rate, a TCAM chip consumes up to 30 times more power than an equivalent SRAM chip with a software-based solution [8]. As a consequence, in the Cisco CRS-1 core router, classification and forwarding cause a third of all power consumption, and constitute the highest source of power together with power management devices such as fans [9].

TCAM devices run each search in parallel on all their entries, therefore their power consumption is about proportional to their number of searched entries. Unfortunately, this number of entries is often larger than the number of classification rules. This is because there are two types of rules: simple rules (exact- and prefix-matches) that need a single entry per rule; and range rules, which might need many entries per rule, thus causing *range expansion*.

TCAM power consumption is increasingly due to this range expansion. Typically, while range rules constitute a minority of the rules, they also cause the majority of the entries, and therefore the majority of the TCAM power consumption [10]. In addition, there is evidence that the percentage of range rules is increasing. For instance, a comparison of two typical classification databases from 1998 and 2004 shows that the total percentage of range rules has increased from 1.3% to 13.3%, including an emergence of rules with two range fields from 0% to 1.5%, and an increase in the number of different ranges [11]. Unfortunately, as the number of range rules increases in an unpredictable way, it is unclear whether it is possible to provide any reasonable guarantee on the worst-case number of TCAM entries needed to encode them.

The goal of this paper is to gain a more fundamental understanding of the worst-case number of TCAM entries needed to encode a rule. Our objective is to provide upper bounds on the worst-case rule expansion, which would characterize the theoretical capacity of TCAM devices depending on the complexity of the rules: e.g., single-field or multiple-field range rules, using simple or complex ranges, either alone or in interaction with other rules. In a sense, we want to help define the *TCAM capacity region*.

B. Related Work

It is well-known that each range defined over a W-bit field can be encoded in $2W-2$ entries for $W \geq 2$ with an *internal* expansion, i.e. an expansion that only uses entries from within the range [12]. More generally, the Cartesian product of d ranges defined on d different fields of length W each can be internally encoded in up to $(2W-2)^d$ entries, which amounts to 900 TCAM entries for $d = 2$ IPv4 port fields of 16 bits each [1].

For instance, assume that $W = 3$ and that we want to internally encode the single-field range $R = [1, 6] \subseteq \left[0, 2^W-1\right]$ so that only packets in R are accepted, while all other packets are denied (default action). Then an internal expansion yields the following $2W-2 = 4$ TCAM entries, not counting the last

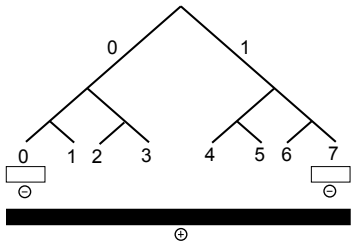


Fig. 1. External encoding of $R = [1, 6]$.

default entry ($*\,*\,* \rightarrow$ deny):

$$\begin{pmatrix} 001 & \rightarrow & \text{accept} \\ 01* & \rightarrow & \text{accept} \\ 10* & \rightarrow & \text{accept} \\ 110 & \rightarrow & \text{accept} \end{pmatrix}$$

A first improvement of the $2W-2$ result has relied on non-prefix internal TCAM encoding and a connection to Boolean Disjunctive Normal Form (DNF) to show a $2W-4$ upper-bound [13]. A second improvement has kept prefix encoding but relied on Gray codes instead of binary codes to reduce the worst-case internal range expansion from $2W-2$ to $2W-4$ for sufficiently large W. This result has since been improved to $2W-5$ using a more complex coding [14]. In any internal coding, the worst-case range expansion is also shown to be at least W [10].

These results, however, do not consider the full potential of TCAM encoding, and in particular the *order of the entries*. For instance, Fig. 1 shows how the example above could be encoded in only 3 TCAM entries using an *external* encoding that exploits a different entry order.

$$\begin{pmatrix} 000 & \rightarrow & \text{deny} \\ 111 & \rightarrow & \text{deny} \\ *\,*\,* & \rightarrow & \text{accept} \end{pmatrix}$$

We can see that the range exterior (complementary) is encoded first, and then the range itself is encoded indirectly. Likewise, in this paper, we consider all possible TCAM entry orders when providing worst-case bounds.

Finally, there is a large literature on providing *efficient heuristics* to minimize the TCAM rule expansion, relying for example on redundancy removal, truth table equivalency, additional bits, additional TCAM hardware, dynamic programming, and topological transformation [1]–[3], [8], [15]–[18]. However, our main objective is to analyze *worst-case* rule expansion guarantees, instead of focusing only on typical average-case performance, even though we also later consider real-life classification databases.

C. Contributions

This paper investigates worst-case rule expansions in TCAMs.

We first consider single-field ranges of W-bit elements, and encode them using efficient algorithms with guaranteed upper bounds. We start by considering W-bit extremal ranges of the form $[0, x]$, and prove that they can be encoded in $g(W) \leq \lceil \frac{W+1}{2} \rceil$ TCAM entries, nearly half the best-known bound of W entries [10].

Later, we consider general ranges of the form $[x_1, x_2]$, and prove that they can always be encoded in $f(W) \leq W$ TCAM entries. Therefore, for large W, this is nearly half the size of the best-known binary bound for prefix TCAM encoding of $2W-2$ and best-known overall bound of $2W-5$ [12], [14].

Then, we prove that any union of k ranges of W-bit elements can be encoded in at most kW TCAM entries.

Last, we propose a modified TCAM architecture that uses additional logic to significantly reduce the rule expansions, with a bound that is *linear instead of exponential* in the number of fields. We explain that hot classifier updates can be applied easily in this architecture. We conclude by illustrating its results both in the worst case and using real-life classification databases.

Note that the result $f(W) \leq W$ has also been independently found in [19], which is presented in the same venue.

Due to space limits, all the proofs are fully presented in [20].

II. Model and Notations

A. Terminology

We first formally define the terminology used in this paper. For simplicity we do not distinguish between a W-bit binary string (in $\{0,1\}^W$) and its value (in $[0, 2^W-1]$).

Definition 1 (Header): A packet *header* $x = (x_1, \ldots, x_d) \in \{0,1\}^W$ is a W-bit string defined on the d fields $(F_1, \ldots, F_d)$. Each sub-string x_i of length W_i corresponds to field F_i, with $\sum_{i=1}^{d} W_i = W$.

Example 1: An IPv4 header consists of the following $d = 5$ fields: $(F_1, \ldots, F_5)$ = (source IP address, destination IP address, source port number, destination port number, and protocol type) of respective lengths $(W_1, \ldots, W_5) = (32, 32, 16, 16, 8)$ bits.

Definition 2 (Range Rule): The *range rule* R_i in field F_i is defined as an integer range $[r_1, r_2]$, where r_1 and r_2 are W_i-bit integers and $r_1 \leq r_2$. A packet header sub-string $x_i \in \{0,1\}^{W_i}$ is said to match R_i whenever $x_i \in [r_1, r_2]$.

In particular, R_i is a *prefix rule* if it consists of all the strings sharing a given prefix. It is an *exact match* if it corresponds to a single string, i.e. $r_1 = r_2$.

Definition 3 (TCAM entry): A *TCAM entry* $S \rightarrow a$ is composed of a TCAM rule $S = (s_1, \ldots, s_W) \in \{0, 1, *\}^W$, where $\{0,1\}$ are bit values and $*$ stands for *don't-care*, and an action $a \in \mathcal{A}$, where $\mathcal{A}$ is a set of actions. A W-bit string $b = (b_1, \ldots, b_W)$ matches S, denoted as $b \in S$, if for all $i \in [1, W]$, $s_i \in \{b_i, *\}$.

Definition 4 (TCAM Encoding Scheme): Consider a function $\alpha : \{0,1\}^W \rightarrow \mathcal{A}$ that associates an action to each packet header. Then a *TCAM encoding scheme* ϕ maps α to an ordered set of $n_\phi(\alpha)$ TCAM entries $\left(S_1 \rightarrow a_1, \ldots, S_n \rightarrow a_{n_\phi(\alpha)}\right)$ and a default action $a_d \in \mathcal{A}$ if and only if for any header $x \in \{0,1\}^W$, $\alpha(x) = a_j$, where a_j is the action associated with the first TCAM entry that matches x, and $a_j = a_d$ if no TCAM entry matches x. The number $n_\phi(\alpha)$ of TCAM entries

with non-default action is the *expansion* of encoding scheme ϕ for classifier function α. Let Φ denote the set of TCAM encoding schemes.

As an example, in Fig. 1, $\mathcal{A} = \{accept, deny\}$, $\alpha(1) = ... = \alpha(6) = accept$, $\alpha(0) = \alpha(7) = deny$. In the remainder of the paper, unless mentioned otherwise, we will assume for simplicity that we only have two actions, with $\mathcal{A} = \{0, 1\}$ and default action $a_d = 0$. Each single-field *range* R is uniquely characterized by its range indicator function α_R, which takes a value of 1 on R and 0 outside R. We will use *range* to indicate either R or its indicator function α_R.

B. Optimal Range Expansion Problem

We want to find a TCAM encoding scheme $\phi \in \Phi$ that minimizes the worst-case TCAM expansion $n_\phi(\alpha_R)$ over all possible ranges R.

Definition 5 (Range Expansion): For any positive integer W and any TCAM encoding scheme $\phi \in \Phi$, the *range expansion* $f_\phi(W)$ of ϕ is the worst-case TCAM expansion $n_\phi(\alpha_R)$ over all possible ranges R, i.e.

$$f_\phi(W) = \max_{R \subseteq [0, 2^W - 1]} n_\phi(\alpha_R). \tag{1}$$

Our goal is to optimize the range expansion over all possible encoding schemes $\phi \in \Phi$. Then the *range expansion* $f(W)$ is defined as the best-achievable range expansion for W-bit ranges given all encoding schemes $\phi \in \Phi$, i.e.

$$f(W) = \min_{\phi \in \Phi} (f_\phi(W)) \tag{2}$$

To find the range expansion $f(W)$, we will first characterize the *extremal range expansion* $g(W)$, which is defined in the same way over the following set of extremal ranges.

Definition 6 (Extremal Ranges): A range R over $[0, 2^W - 1]$ is called extremal if $R = [0, y]$ or $R = [y, 2^W - 1]$ for some arbitrary value of y.

III. Range Expansion Guarantees

A. Upper-Bound on the Extremal Range Expansion

We are now ready to characterize the extremal range expansion $g(W)$. We will improve the best-known upper bound on $g(W)$ from W [10] to $\left\lceil \frac{W+1}{2} \right\rceil$. To do so, we rely on external encoding, i.e. we will sometimes first encode negatively the complementary range, as illustrated in the following example.

Example 2: For $W = 3$, the extremal range $R = [0, 6]$ can be encoded with only $\left\lceil \frac{W+1}{2} \right\rceil = \left\lceil \frac{3+1}{2} \right\rceil = 2$ TCAM entries $(111 \rightarrow 0, *** \rightarrow 1)$. This is better than the best internal encoding $(0** \rightarrow 1, 10* \rightarrow 1, 110 \rightarrow 1)$, which uses $W = 3$ TCAM entries.

Theorem 1: For all $W \geq 1$, the extremal range expansion satisfies the following upper-bound:

$$g(W) \leq \left\lceil \frac{W+1}{2} \right\rceil \tag{3}$$

B. Upper-Bound on the Range Expansion

We now want to find an upper-bound on the range expansion $f(W)$ by constructing an efficient encoding scheme. As for $g(W)$, external encoding improves the best-known upper bound on $f(W)$ by a factor of about 2, from $2W - 5$ [14] to W.

Theorem 2: For all $W \geq 1$, the worst-case range expansion satisfies the following upper-bound:

$$f(W) \leq W. \tag{4}$$

In fact, as proved in [20], this upper bound still holds even when constraining the encoding to be a prefix encoding.

C. Union of ranges

In spite of the last result, it is not straightforward that encoding k ranges would also be possible in kW TCAM entries. For instance, if we encode some range R_1 using external encoding, i.e. by first encoding negatively its complementary R_1^c, we might encompass another range $R_2 \subseteq R_1^c$, and therefore yield a wrong encoding. By considering all the range combinations and distinguishing extremal from non-extremal ranges, we prove that the result is actually correct.

Theorem 3: For all $W \geq 1$ and $k \in \mathbb{N}$, any k ranges of W-bit elements can be encoded in at most kW TCAM entries.

IV. Multidimensional ranges

A. Exponential Number of TCAM Entries

We now want to encode a multidimensional rectangle, i.e. a hyper-rectangle defined as a Cartesian product of the one-dimensional intervals that we encoded above. We will show that a more efficient encoding of the intervals leads to a more efficient encoding of the hyper-rectangle.

Example 3: Consider a general range of two W-bit fields $R = (R_1, R_2)$, presented in Fig. 2(a). For $i = 1, 2$ let r_i be the expansion of R_i using internal encoding, and let r'_i be the expansion of R_i using our improved encoding scheme. It is well-known that R can be encoded with $r_1 \cdot r_2 \leq (2W-2)^2$ TCAM entries. Likewise, it can be encoded with $r'_1 \cdot r'_2 \leq f(W)^2 \leq W^2$ TCAM entries.

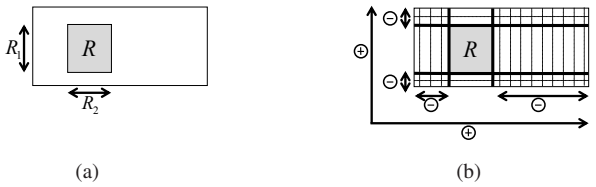
More generally, given a hyper-rectangle with d dimensions, we get:

Theorem 4: For any classification rule $R = ((R_1, \ldots, R_d) \rightarrow a)$ of d fields, there is an encoding scheme ϕ s.t. $n_\phi(\alpha_R) \leq W^d$.

B. Linear Number of TCAM Entries

The main drawback of encoding a hyper-rectangle with d dimensions is the curse of dimensionality, i.e. the typical exponential dependency in the number of fields d. We show here how to encode a hyper rectangle with a linear dependency in d.

Example 4: Consider again the range R from Example 3. As illustrated in Fig. 2(b), we can first negatively encode the four striped regions, using an encoding of the corresponding four one-dimensional extremal intervals (using $4W$ entries [12]), and then add a default positive entry (using one entry), thus yielding a linear expansion upper-bound of $4W + 1$.



(a) (b)

Fig. 2. Two-dimensional range $R = (R_1, R_2)$

More generally, we get the following tighter upper-bound:

Theorem 5: Any classification rule R of d fields can be encoded in at most $d \cdot (2W - 2) + 1$ TCAM entries without any additional logic.

C. Linear Number of TCAM Entries with Additional Logic

The above two multidimensional results assume that the classifier has only one classification rule. In Section V we suggest hardware changes that enable us to efficiently encode $k > 1$ classification rules, as stated in the following theorem.

Theorem 6: Let $C = (R^1, ..., R^k)$ be a classifier with k classification rules defined over d fields. Using additional logic, C can be encoded in at most $k \cdot d \cdot W$ TCAM entries.

V. TCAM Architectures

A. Suggested Architectures

In this section we suggest several TCAM architectures that enable us to implement range encoding more efficiently using logical gates, and illustrate them with a simple example. These TCAM architectures trade off better range expansions, and therefore less TCAM entries, against more complex logic within the TCAM. Note that the use of logic gates to process TCAM results is not generally new [16], [17], but these logic-based architectures apparently are.

Fig. 3 illustrates the different TCAM architectures. We assume $d = 2$ fields of $W = 4$ bits each. We want to encode $k = 2$ multi-dimensional ranges R^1 and R^2, where $R^1 = [1, 14] \times [5, 14]$, $R^2 = [7, 10] \times [2, 3]$, and each range leads to a different action. We assume that the default action is predefined in the priority encoder (PE), and therefore there is no need to add TCAM entries for it. We also use an input packet-header example $x = (8, 7) = (1000, 0111)$. Note that x is in R^1, but not in R^2. We denote in parentheses the values that are transmitted on each line.

First, Fig. 3(a) presents the standard INTERNAL-PRODUCT architecture. As usual, using internal binary-prefix encoding, it encodes each range by using the product of its TCAM entries along each dimension. In this case, it uses 6×5 entries to encode R^1, and 3×1 entries to encode R^2, yielding a total of 33 entries.

Next, Fig. 3(b) introduces the proposed COMBINED-PRODUCT architecture. Instead of encoding each range only internally, the COMBINED-PRODUCT architecture encodes it using its complementary as well, in at most $f(W) = W$ entries instead of $2W-2$. It also uses more logic to process the results. In this example, it uses 12 entries for R^1 and 3 for R^2, i.e. a total of 15.

Specifically, each field of each range behaves like a single TCAM block. The results of each TCAM entry are entered into a chained logic part that outputs a (1) on each line if it is the first entry that matches the header, and (0) otherwise (i.e. either there was no match on this line or there was a match on a previous line). Note that the chained logic can also be replaced with a more efficient hierarchical logic.

In the second logic part, a logic gate with a control input either behaves like a pass-through gate or like a zeroing gate (drawn as a box), depending on whether the encoded entry corresponds to the range or to its complement. Thus, the output is a (1) if and only if it is the first matching entry and it belongs to the range. Last, an OR gate checks whether the first matching entry belongs to the range, i.e. whether the range is matched. The PE then outputs the first matching range.

Next, Fig. 3(c) shows the proposed COMBINED-SUM architecture. Each field of each range is encoded separately, by using chaining as in the COMBINED-PRODUCT architecture. Then, in a second stage, an AND gate checks whether all fields have a match. In this example, R^1 is encoded using $3+4 = 7$ entries, and R^2 using $3 + 1 = 4$, with a total of 11 entries.

Table I summarizes the bounds on the worst-case rule expansion for each architecture, and provides the corresponding values for IPv4 packet headers with 2 range fields of 16 bits each.

TABLE I
UPPER BOUND ON RULE EXPANSIONS OF TCAM ARCHITECTURES

Architecture	Expansion upper bound	Values for $k = 1$, $W = 16$, $d = 2$
INTERNAL-PRODUCT	$k \cdot (2W-2)^d$	$(30)^2 = 900$
COMBINED-PRODUCT	$k \cdot W^d$	$(16)^2 = 256$
COMBINED-SUM	$k \cdot d \cdot W$	$2 \cdot 16 = 32$

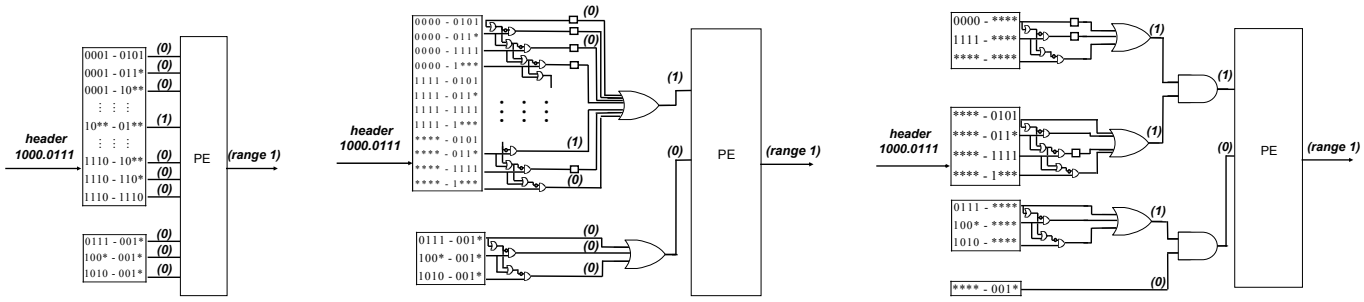
B. Implementation Considerations

Hot Updates: Since the TCAM is clearly divided between ranges, and the implementation of each range is independent of the other ranges, hot classifier updates are surprisingly easy to apply in this architecture compared to typical TCAM architectures.

Turning Off Entries: In the figures, we only represent the active entries. A simple way to implement the TCAM is to divide it by blocks, each block representing the maximum number of entries per range (Table I). Then, when some entries are not used, it is possible to turn them off. To do so, add a transistor to switch voltage on and off, together with an SRAM array of 1 bit per entry that remembers the correct action.

PE Size: The number of inputs and outputs of the PE is reduced. It is now the number of range rules, instead of the number of TCAM entries. In a sense, the PE is implemented in a hierarchical fashion, with the first logic block being the one shown in the figure.

Multiple Actions: To implement more actions than *accept* and *deny*, the architecture does not need to be changed. The



(a) INTERNAL-PRODUCT (b) COMBINED-PRODUCT (c) COMBINED-SUM

Fig. 3. TCAM Architectures

TABLE II
AVERAGE EXPANSION RATIOS WITH REAL-LIFE CLASSIFIERS.

Parameters	All rules	1 range-field	2 range-fields
Fraction of all rules (%)	*100%*	*26%*	*1.5%*
INTERNAL-PRODUCT	2.68	7.32	47.18
COMBINED-PRODUCT	1.63	3.38	20.09
COMBINED-SUM	2.45	3.69	8.80

action associated with each range simply needs to be indicated in the corresponding SRAM entry.

More implementation details can be found in [20].

VI. EXPERIMENTAL RESULTS

We evaluated the suggested architectures on a real-life database of 120 separate rule files and about $215,000$ rules originating from various applications (such as firewalls, ACL-routers and intrusion prevention systems). The database was previously used in [1], [10], [11], [21].

As shown in Table II, the COMBINED-PRODUCT architecture outperformed the other architectures. With respect to the baseline INTERNAL-PRODUCT architecture, it improved by 39% the total number of TCAM entries, and in particular by 54% and 57% the number of TCAM entries needed for one and two range-fields, respectively. The COMBINED-SUM architecture also performed slightly better than the baseline architecture.

For additional simulations, please refer to [20].

VII. CONCLUSION

This paper is unique in that it deals with the fundamental capacity region of TCAMs. In the paper, we presented new upper-bounds on the TCAM worst-case rule expansions. In particular, we proved that a W-bit range can be encoded in W TCAM entries using prefix encoding, improving upon the previously-known bound of $2W-5$.

In addition, we suggested several modified TCAM architectures that can trade off better range expansions, and therefore less TCAM active entries, against more complex logic within the TCAM. Last, we showed that it is possible to encode ranges using a number of TCAM entries that increases only *linearly* instead of *exponentially* with the number of fields.

ACKNOWLEDGMENT

This work was partly supported by the European Research Council Starting Grant no. 210389. We would like to acknowledge Anat Bremler-Barr for kindly accepting to run several simulations. We would also like to thank David Hay, Danny Hendler, Ran Ginosar and Zigi Walter for their helpful suggestions.

REFERENCES

[1] D. E. Taylor, "Survey and taxonomy of packet classification techniques," *ACM Comput. Surv.*, vol. 37, no. 3, pp. 238–275, 2005.
[2] G. Varghese, *Network Algorithmics*. Morgan Kaufmann, 2005.
[3] J. Chao and B. Liu, *High Performance Switches and Routers*. Wiley, 2007.
[4] J. Naous *et al.*, "Implementing an OpenFlow switch on the NetFPGA platform," in *ACM ANCS*, 2008.
[5] NetLogic Microsystems. [Online]. Available: www.netlogicmicro.com/
[6] Renesas. [Online]. Available: www.renesas.com/
[7] K. Pagiamtzis and A. Sheikholeslami, "Content-addressable memory (CAM) circuits and architectures: A tutorial and survey," *IEEE J. Solid-State Circuits*, vol. 41, no. 3, pp. 712–727, Mar. 2006.
[8] C. R. Meiners, A. X. Liu, and E. Torng, "Bit weaving: A non-prefix approach to compressing packet classifiers in TCAMs," Michigan State University, Technical Report MSU-CSE-09-1, Jan. 2009.
[9] J. Baliga *et al.*, "Photonic switching and the energy bottleneck," in *Photonics in Switching*, Aug. 2007, pp. 125–126.
[10] A. Bremler-Barr and D. Hendler, "Space-efficient TCAM-based classification using Gray coding," in *IEEE Infocom*, 2007, pp. 1388–1396.
[11] K. Lakshminarayanan, A. Rangarajan, and S. Venkatachary, "Algorithms for advanced packet classification with ternary CAMs," in *ACM SIGCOMM*, 2005, pp. 193–204.
[12] V. Srinivasan *et al.*, "Fast and scalable layer four switching," in *ACM SIGCOMM*, 1998, pp. 191–202.
[13] B. Schieber, D. Geist, and A. Zaks, "Computing the minimum DNF representation of Boolean functions defined by intervals," *Discrete Applied Mathematics*, vol. 149, no. 1-3, pp. 154–173, 2005.
[14] R. Roth, Personal communication.
[15] C. R. Meiners, A. X. Liu, and E. Torng, "TCAM razor: A systematic approach towards minimizing packet classifiers in TCAMs," in *ICNP*, 2007, pp. 266–275.
[16] E. Spitznagel, D. E. Taylor, and J. S. Turner, "Packet classification using extended TCAMs," in *ICNP*, 2003, pp. 120–131.
[17] H. Hwang *et al.*, "Minimization of ACL storage by adding minimal hardware of range matching and logical gates to TCAM," in *HSPR*, 2008.
[18] S. Suri, T. Sandholm, and P. R. Warkhede, "Compressing two-dimensional routing tables," *Algorithmica*, vol. 35, no. 4, pp. 287–300, 2003.
[19] R. Cohen and D. Raz, "Simple efficient TCAM based range classification," in *IEEE INFOCOM Mini-Conference*, 2010.
[20] O. Rottenstreich and I. Keslassy, "Worst-case TCAM rule expansion," Comnet, Technion, Israel, Technical Report TR09-01, 2009.
[21] A. Bremler-Barr *et al.*, "Layered interval codes for TCAM based classification," *IEEE Infocom*, 2009.