This paper was presented as part of the Mini-Conference at IEEE INFOCOM 2011

Multi-Field Range Encoding for Packet Classification in TCAM

Yeim-Kuan Chang, Chun-I Lee and Cheng-Chien Su
Department of Computer Science and Information Engineering
National Cheng Kung University, Taiwan, R.O.C.

***Abstract*—Packet classification has wide applications such as unauthorized access prevention in firewalls and Quality of Service supported in Internet routers. The classifier containing pre-defined rules is processed by the router for finding the best matching rule for each incoming packet and for taking appropriate actions. Although many software-based solutions had been proposed, high search speed required for Internet backbone routers is not easy to achieve. To accelerate the packet classification, the state-of-the-art ternary content-addressable memory (TCAM) is a promising solution. In this paper, we propose an efficient multi-field range encoding scheme to solve the problem of storing ranges in TCAM and to decrease TCAM usage. Existing range encoding schemes are usually single-field schemes that perform range encoding processes in the range fields independently. Our performance experiments on real-life classifiers show that the proposed multi-field range encoding scheme uses less TCAM memory than the existing single field schemes. Compared with existing notable single-field encoding schemes, the proposed scheme uses 12% ~ 33% of TCAM memory needed in DRIPE or SRGE and 56% ~ 86% of TCAM memory needed in PPC for the classifiers of up to 10k rules.**

Keywords-TCAM, Packet classification, multi-field range encoding

I. INTRODUCTION

In modern network architecture, routers are the most important components. A router is a device that interconnects two or more networks and interchanges packets between them. By inspecting the information in the packet header, routers can decide the target network and select the preferred path between any two networks for the packets. However, the rapid growth of Internet has caused increasing congestion and packet loss at intermediate routers in recent years. Internet service providers (ISPs) would like to provide the differentiated services. Therefore, some important new network services are developed for routers to provide different levels of services. To meet the service requirements, routers need to implement a new function, called packet classification, to distinguish and classify the incoming packets into different classes of services.

Packet classification is an enabling function in routers to support many network applications, such as Quality of Service (QoS), security, monitoring, and network intrusion detection. To achieve the high performance, the speed of packet classification is often a bottleneck in routers. To perform the function of packet classification, routers need to recognize the information of the incoming packets specified by a *classifier* containing a set of *rules* that are used to check the header field values. Packet classification is the process of identifying the rules within a classifier that the incoming packet matches. Rules in the classifier consist of five fields and an action value. The five fields are the source/destination IP addresses, the source/destination port numbers, and the protocol number. In order to decide the action taken for each incoming packet, the router needs to search the matching rule in the classifier.

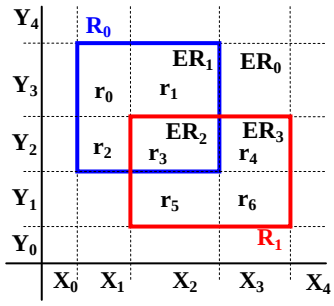
With the increasing network traffic and size of classifiers, packet classification speed is becoming more and more important. In recent years, many software-based packet classification schemes are proposed [4][9][11][19], but they are not fast enough to reach the performance demanded by Internet backbone routers. To accelerate the search speed, special hardware support is a good approach. Ternary content-addressable memory (TCAM) is often used to solve the packet classification problem because of its speed, simple design and management. When a search operation is undertaken in TCAM, parallel comparisons on all TCAM entries against the input data are processed and all matching entries can be output in one clock cycle. Another feature is that TCAM allows a third matching state of "*" or "don't care". If a bit is set to '*', it can be matched by "0" and "1".

Although TCAM can compare all entries in one clock, it still has three primary disadvantages that are high hardware cost, high power consumption, and inefficiency in storing range data. In order to store rules into TCAM, the issue of storing range data such as source and destination port numbers must be solved. Any arbitrary range can be pre-processed to convert to one ore more ternary strings which contain "don't care" bits. This preprocessing procedure is called *range encoding*. To store rules into TCAM, the source and destination port field values should be encoded. Finally, the ternary strings obtained by encoding the range field values are concatenated with other three prefix fields before being put into TCAM. Because the length of ternary strings and the number of concatenations greatly affect the TCAM memory usage, how to design a memory-efficient encoding scheme is the main issue in this paper.

The rest of the paper is organized as follows. In section II, the related work for packet classification is briefly described. The section III illustrates the proposed schemes. Section IV experimental results and the last section concludes the paper.

II. RELATED WORK

The encoding scheme for packet classification can be categorized into two types, *database-independent* and *database-dependent* schemes. For database-dependent schemes,



EI_x, EI_y	Region	ER*i*	codeword
(X_0, Y_0)	-	ER_0	00
(X_1, Y_2)	r_2	ER_1	01
(X_1, Y_3)	r_0	ER_1	01
(X_2, Y_1)	r_5	ER_3	10
(X_2, Y_2)	r_3	ER_2	11
(X_2, Y_3)	r_1	ER_1	01
(X_3, Y_1)	r_6	ER_3	10
(X_3, Y_2)	r_4	ER_3	10
Other regions are omitted because they are the same as (X_0,Y_0). R0 and R1 are encoded as ternary strings *1 and 1*, respectively.			

Fig. 1: 2-D original ranges, regions, elementary regions, and coodewords.

the codeword assignment of a range is not independent of other ranges. While performing search operations, the router needs to fetch the codeword corresponding to the range search key from memory first, and then uses the codeword to execute the matching operation in TCAM. On the contrary, database-independent encoding schemes do not need additional memory to store codewords, and each range can be encoded independently. The advantage of database-dependent encoding schemes is the efficiency of utilizing memory space. But, the drawback is that it is hard to perform update operations when a rule is added or deleted because all codewords need to be re-calculated. Subsequently, we will briefly describe some famous database-independent and database-dependent schemes.

A. Database-independent Range Encoding

In order to encode arbitrary range independently from other ranges, the *direct range-to-prefix conversion* [3] is the simplest scheme that uses multiple prefixes to represent a range. But, its worst case is 2W-2 prefixes for a range in W-bit address space. In the direct conversion, Gray code is better than binary Buddy code for encoding ranges into ternary strings. Two successive codewords in Gray code must be differed by one bit and thus, fewer ternary strings are needed for a range than Buddy code. The *Overlapping Range Encoding* (ORE) [8] and *Short Range Gray code Encoding* (SRGE) [1] provide the efficient way to find the near minimum number of ternary strings for a range. In [12], authors proposed another scheme, called *Database Independent Range PreEncoding* (DIRPE). By using specific format to represent a value, DIRPE can convert the range to fewer ternary strings directly. *TCAM Razor* [16] and *Range Code-Length Optimality* [17][18] reduce the number of TCAM entries by identifying semantically equivalent rule sets.

B. Database-dependent Range Encoding

The *Bitmap-intersection* scheme [14] is a straightforward database-dependent scheme in which each port range corresponds to a bit of a bitmap used to record the covering rules. But the disadvantage is the size of bitmap is dependent on the number of distinct ranges. To solve this problem, the elementary interval based encoding schemes are proposed. The *elementary interval-based scheme using binary reflected Gray code* (EIGC) [6] scheme assigns each elementary interval a codeword based on Gray code [10]. The ternary strings for a range can be obtained by combining the codewords of all the elementary intervals covered by the range. Another encoding scheme called *Parallel Packet Classification* (PPC) [15] groups all rules into layers and each layer can be performed encoding scheme independently. Because too many layers will cause longer codeword, the *Layered Interval Encoding* scheme [2] provides methods to find the maximum independent rule set. There is another type of encoding scheme, called *hybrid encoding scheme*, such as DRES [7]. The main idea is that the extra TCAM bits are used to encode the rules which cause large rule expansion and thus the encoding complexity can be decreased.

III. PROPOSED SCHEME

In this section, we propose a multi-field range encoding algorithm. We process multiple fields simultaneously and assign suitable ternary strings for all the two-field ranges where the two fields are assumed to be source and destination port ranges in this paper. In most cases, the length of ternary string in multi-field encoding scheme is shorter than that of single-field encoding scheme. In order to decrease the TCAM memory usage, our proposed scheme solves this problem by using one TCAM entry for each rule and the length of the ternary string can be limited.

The two-field range defined in a rule is called *original 2-D range* in the paper. Before introducing the proposed encoding algorithms, the following definitions of region and elementary region are needed. The relationships of two original 2-D ranges must satisfy one of following three conditions:

1) Disjoint: A and B are disjoint if and only if address intersection of A and B is empty, i.e., $A \cap B = \varnothing$.

2) Partially overlapped: A is partially overlapped with B if and only if $A \cap B \neq \phi$ or A or B.

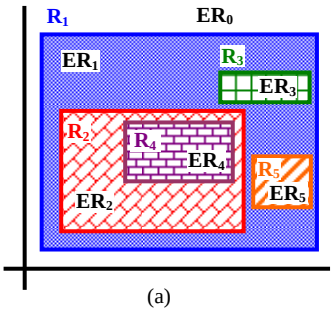
3) Enclosed: A encloses B if and only if $A \cap B = B$.

***Definition 1**: A **region** is a rectangular area corresponding to a pair of 1-D elementary intervals, which is composed from the source and destination port range fields.*

Fig. 1 shows a simple example. There are two overlapping original 2-D ranges R_0 and R_1. Five elementary intervals X_0 to X_4 in field X and five elementary intervals Y_0 to Y_4 in field Y are formed from these two rules. As a result, there are 5*5 = 25 rectangular regions each of which corresponds to a pair of two elementary intervals belonging to fields X and Y. For instance, region r_1 in Fig. 1 is formed by elementary interval pair (X_2, Y_3). Original 2-D range R_0 contains four regions r_0, r_1, r_2, and r_3, and Original 2-D range R_1 contains four regions r_3, r_4, r_5, and r_6.

***Definition 2 (Elementary region)**: Let the set of k elementary regions constructed from an original 2-D range set R of 2-D W-bit rules be $X = \{ER_i \mid i = 1 \text{ to } k\}$. Each elementary region ER_i covers a subset of addresses in the 2D address space of $(0 \ldots 2^W - 1, 0 \ldots 2^W - 1)$. X must satisfy the following: (1) All addresses in ER_i are covered by the same subset of original 2-D ranges (called the range matching set of ER_i, denoted by ER_i.range), and (2) The range matching sets of two different elementary regions are not equivalent.*

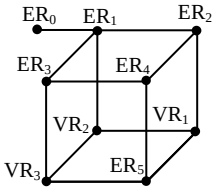
Based on above definition, similar to elementary interval defined in [5], the regions belonging to the same elementary region match the same set of original 2-D ranges. The shape of an elementary region is not necessarily a rectangular and also does not necessarily cover a contiguous address space.



(a)

R_i	$Vset_i$
R_1	ER_1, ER_2, ER_3 ER_4, ER_5, VR_1, VR_2, VR_3
R_2	ER_2,ER_4
R_3	ER_3
R_4	ER_4
R_5	ER_5

(b)



(c)

Codeword memory	
ER*i*	Code
ER_0	0000
ER_1	1000
ER_2	1001
ER_3	1010
ER_4	1011
ER_5	1111
VR_1	1101
VR_2	1100
VR_3	1110

(d)

TCAM	
R*i*	*Ternary String*
R_1	1***
R_2	10*1
R_3	1010
R_4	1011
R_5	1111

(e)

Fig 2: (a) The original 2-D range set in one layer. (b) Vset for each 2-D ranges. (c) encoding sub-cube in 4-cube. (d) ER*i*'s codeword. (e) R*i*'s ternary string.

Consider the same example in Fig. 1. There are four elementary regions constructed from the original 2-D range R_0 and R_1. Elementary region ER_1 covers regions r_0, r_1, and r_2, ER_2 covers region r_3, and ER_3 covers regions r_4, r_5, and r_6. ER_0 covers all the remaining regions. The search operation must locate the elementary region corresponding to the header field values of the incoming packet and return the intermediate codeword of the located elementary region which is then used to search the ternary strings constructed from the proposed 2-D range encoding schemes.

In a single-field encoding scheme, such as PPC, the intermediate codewords have to be assigned to all elementary intervals and also the ternary strings for all the original 1-D ranges have to be determined. For the same reason, in our proposed scheme, we need to assign intermediate codewords to all elementary regions. For example, ER_1 in Fig. 1 is assigned codeword "01", and ER_2 is assigned codeword "11", and ER_3 is assigned codeword "10". By combining the codewords of elementary regions ER_1 and ER_2, R_0 can be expressed as a ternary string "*1", Similarly, R_0 can be expressed as a ternary string "1*". In addition, the intermediate codeword '00' has to be assigned to the default elementary region. The main issue is how to assign an appropriate codeword of length as short as possible to each elementary region such that each original 2-D range can be represented by only one ternary string. Consider the same example in Fig. 1. Suppose ER_1, ER_2, and ER_3 are assigned with "01", "10", and "11", respectively. R_1 can be represented by one ternary string "1*" but two ternary strings "01" and "10" are needed for R_0. So inappropriate elementary region codeword assignment will fail to represent each original 2-D range as one ternary string, which is the primary objective of the proposed encoding algorithms.

In single-field searching operation, the router needs two memory accesses to fetch the codewords of respective port fields. Then, the two found codewords are concatenated with the header values of other fields to be the searching key in TCAM. The multi-field hardware architecture is similar to the single-field architecture. When an incoming packet arrives, the router fetches port numbers of two port fields from packet headers. By using those two port numbers, the router can find out two corresponding elementary intervals of IDs EI_x and EI_y. Then, IDs EI_x and EI_y are used as the key to search the codeword memory structured as a 2D array using the elementary interval IDs as the indices. Finally, from the codeword memory, the codeword of the corresponding region can be obtained. Based on this procedure, we need two memory accesses (can be run in parallel) to access the elementary interval ID arrays and one memory access to obtain the codeword from codeword memory. Compared to the single-field searching architecture, the multi-field search architecture needs only one more memory access. Although the multi-field search architecture needs additional one memory access, the total SRAM access time is quiet small.

B. *Layered Approach*

If the relationship of any two original 2-D ranges R_i and R_j is disjoint or enclosed, performing the codeword assignment can be as simple as in PPC [15]. We classify all original 2-D ranges into many groups, called *layers* in which the relationship between any two original 2-D ranges in the same layer must be disjoint or enclosed. We can perform the encoding procedure for each layer independently. Unlike PPC scheme, our proposed layered scheme can put the original 2-D ranges into the same layer no matter they are enclosed and disjoint.

Our goal is to assign a codeword to each elementary region, and each original 2-D range can be represented by only one ternary string. We need additional structures and constraints to execute the codeword assignment. In this paper, we use graph theory to find the correct and efficient codeword assignment. The proposed codeword assignment algorithm is based on *hypercube*. Hypercube is suitable for encoding because of its regularity and symmetry properties. An *n*-dimensional hypercube is also called an *n*-cube or Q_n, which contains 2^n vertices, $n2^n$-1 edges, and the degree of each vertex is *n*. The most important property is that each node in an *n*-cube can be uniquely represented by an *n*-bit codeword in such a way that two vertices are adjacent if and only if their codeword differ in exactly one bit. If a graph is a subgraph of an *n*-cube, each vertex can get a codeword from the corresponding vertex in *n*-cube. We will try to convert all original 2-D ranges to a graph and find a mapping from a vertex in the *n*-cube to each elementary region. If it is successful, it means all elementary regions can be assigned with an *n*-bit codeword and ultimately each 2D range can be represented by only one ternary string corresponding to a sub-cube. The vertices mapped to the elementary regions covered by a 2-D range R_i forms a vertex set, called $Vset_i$. The following constraint is the necessary condition to meet for all 2D ranges.

Constraint 1: $|Vset_i| = 2^n$ and $Vset_i$ must form an *n*-cube.

In constraint 1, because any sub-cubes in an *n*-cube can be represented as one ternary string, we restrict the number of vertices in each original 2-D range to be a power of 2. If the

```
R: the original 2-D range set
AdjMatrix: Record all edges for entire graph
01 function Encoding(R, AdjMatrix)
02     Ordered_list = sort R in decreasing order by their Vset sizes
03     while (Ordered_list is not empty)
04         R_i = the first original 2-D range in Ordered_list
05         add virtual regions for R_i
06         Map-To-Cube(R_i, AdjMatrix)
07         remove the first original 2-D range from Ordered_list
08     end while
09     Assign-Codeword(AdjMatrix)
10 end function
11 function Map-To-Cube(R_i, AdjMatrix)
12     d = ⌈log2(# of elementary regions and virtual regions in R_i)⌉
13     Create a d-dimensional sub-cube Q_sub
14     Obtain all sub-cubes which belong to R_i from AdjMatrix
15     Find mappings for all sub-cubes and isolated vertices in Q_sub.
16     Record all edges of Q_sub in AdjMatrix
17 end function
18 function Assign-Codeword (AdjMatrix)
19     Create a d-dimensional Q, where d = ⌈log2(# of elementary regions
       and virtual regions)⌉
20     Obtain all sub-cubes from AdjMatrix
21     Find mappings for all sub-cubes and isolated vertices in Q.
22     Assign codewords to all elementary regions according to the
       corresponding vertices in Q
23 end function
```

Fig 3: The pseudo code of layered encoding scheme.

produced graph is a sub-graph of an *n*-cube, every elementary region can be assigned an appreciate codeword, and the ternary string of each original 2-D range can be obtained by combining all the codewords of the sub-cube. After converting all the original 2-D ranges to a graph, if we show that the converted graph is a sub-graph of an *n*-cube, we can easily carry out the process of codeword assignment.

For the purpose of finding the correct result, complying with constraint 1 is necessary. If there is one or more original 2-D ranges not complying with constraint 1, it is impossible to find the correct result. Fig. 2(a) shows an example and Fig. 2(b) list the *Vsets* of all original 2-D ranges. It is obvious that the $Vset_1$ of R_1 does not comply with constraint 1 because R_1 contains 5 elementary regions.

In order to resolve this problem, we add extra elementary regions (also called virtual regions) to satisfy constraint 1. Because virtual regions are fictitious, they will not be matched against input key. After adding virtual regions to some elementary regions, all original 2-D ranges can conform to constraint 1. Because we also need to assign a codeword to every virtual region, producing too many virtual regions will increase the complexity of finding the mapping of a graph onto a sub-cube. Thus, we have to limit the number of virtual regions added as much as possible. In order to find the minimum allocation of virtual regions, we should check all original 2-D ranges in a decreasing order of their Vset size. Assume there are two original 2-D range R_A and R_B in the same layer and the Vset R_A is larger than R_B. We add virtual regions to R_B before R_A because R_A may enclose R_B. Fig. 2(b) shows the result. Because R_2, R_3, R_4, and R_5 satisfy constraint 1, R_1 is appended with three virtual regions VR_1, VR_2, and VR_3.

Another important problem is how to connect the correct edges between the vertices corresponding to all elementary regions. In constraint 1, the elementary regions belonging to the same original 2-D range should form a sub-cube in an *n*-cube. So, the edge-connecting order is important. If the original 2-D range R_A encloses R_B, the edge-connecting procedure should process R_B before R_A. For example, in Fig. 2(c), the edge- connecting order should be in the order of $R_4 \rightarrow R_2$, R_3, or $R_5 \rightarrow R_1$. Because R_2, R_3, and R_5 are mutual disjoint, process those three original 2-D ranges in arbitrary order will not affect the final produced graph.

Fig. 3 shows the pseudo code of the proposed encoding algorithm. In line 2, in order to get the processing order, we record all original 2-D ranges in the decreasing order of their *Vset* sizes. In line 5-6, the original 2-D range R_i must comply with constraint 1, so we add minimum number of virtual regions to R_i so that the size of $Vset_i$ is a power of 2. Then, we use function *Map-To-Cube* for R_i to map the vertices corresponding to all elementary regions and virtual regions of R_i onto a sub-cube. To record the entire graph, we use adjacent matrix *AdjMatrix* to track all created edges, and all edges cannot be modified after being created. Repeat line 3-8 until all original 2-D ranges are processed. In line 9, the function *Assign-Codeword* maps the entire graph onto an *n*-cube, and each elementary region can obtain a codeword from the corresponding vertex in *n*-cube. Because we classify all original 2-D ranges into several layers, each layer can perform the encoding scheme independently, and every region can obtain a codeword by concatenating the codewords of all layers. For a search operation, the router can fetch a codeword from the located region by the port number of two port fields, and find the best matching rule via TCAM.

IV. Experimental Results

We compare the proposed scheme with existing algorithms in terms of TCAM entry size, TCAM size and SRAM size, and perform the experiments with classifiers of various sizes. ClassBench [20] is a well-known benchmark that provides classifiers similar to real classifiers used in the Internet routers and input traces corresponding to the classifiers. The three different type classifiers, *access control lists* (ACL), *firewalls* (FW), and *IP chains* (IPC) are generated by *ClassBench* and experimented in the simulation. Because the proposed schemes are designed for encoding original 2-D range, the source and destination port fields in classifiers are only used in the experiments. In order to evaluate the performance of our proposed schemes for classifiers of different sizes, we use 3 synthetic classifiers which are *fw1*, *acl1*, and *ipc1* with size 10,000. The evaluated schemes are direct range-to-prefix conversion (DC) [3], SRGE [1], EIGC [6], DIRPE [12], PPC [15], and our proposed scheme *layered approach* (Layer).

Table I shows the results for the synthetic classifiers of around 10,000 rules. In order to correctly show the encoding results of single-field encoding schemes with two fields, the TCAM entry size of single-field encoding schemes is obtained by concatenating the encoding results of the two fields. In *acl1_10k* classifier, since the range in source port is only wildcard, the needed TCAM entry size can be decreased in most of the schemes, such as DIRPE, PPC, EIGC, and Bitmap.

TABLE I. PERFORMANCE OF 10K CLASSIFIERS

# of rules	scheme	entry size		# of TCAM entry	EF	TCAM size (kb)	SRAM size (KB)
		src	dest				
9,311 (fw1_10k)	DC	16	16	32,136	3.45	1,004.25	0.00
	SRGE	16	16	32,124	3.45	1,003.88	0.00
	EIGC	5	6	62,779	6.74	674.38	88.00
	DIRPE	34	29	14,838	1.59	912.88	0.00
	PPC	6	8	9,311	1.00	127.30	112.00
	Layer	12		9,311	1.00	109.11	90.95
9,603 (acl1_10k)	DC	16	16	12,947	1.35	404.59	0.00
	SRGE	16	16	12,510	1.30	390.94	0.00
	EIGC	1	7	21,633	2.25	169.01	64.00
	DIRPE	1	29	11,374	1.18	333.22	0.00
	PPC	1	15	9,603	1.00	150.05	128.00
	Layer	9		9,603	1.00	84.4	64.22
9,037 (ipc1_10k)	DC	16	16	12,127	1.34	378.97	0.00
	SRGE	16	16	11,937	1.32	373.03	0.00
	EIGC	6	7	120,193	13.30	1,525.89	104.00
	DIRPE	34	29	10,203	1.13	627.72	0.00
	PPC	9	13	9,037	1.00	194.15	176.00
	Layer	15		9,037	1.00	132.38	115.30

The number of TCAM entries is the sum of the required entries for all converted ranges, and the value of EF is the expansion ratio which is calculated by (# of TCAM Entries) / (# of Rules). In some schemes such as DC and EIGC, the expansion ratio is larger than other schemes because most rules require more than one ternary strings or prefixes after conversion. The TCAM size is calculated by (# of TCAM Entries * TCAM Entry size) / 1024, and the SRAM size in Kbyte of single-field range encoding scheme is calculated by (65536 * sum of entry sizes in source and destination port) / (8×1024). In our proposed schemes, we need to count the number of elementary interval in each field. Assume there are *s* elementary intervals in source port field and *d* elementary intervals in source port field. The SRAM size is calculated by (65536*($\lceil \log_2 s \rceil + \lceil \log_2 d \rceil$) + *s* * *d* * the size of TCAM entry) / (8×1024).

It is obvious that DC has the worst performance because it requires the largest number of TCAM entries and the size of TCAM entry is 32 bits. PPC, DIRPE and our proposed schemes need the minimum number of TCAM entries because those schemes are database-dependent encoding schemes, which usually focus on decreasing expansion ratios. Comparing the entry size with all schemes, the proposed schemes can use the minimum TCAM cost in all classifiers, and the SRAM usage are also better than single-field encoding schemes.

V. CONCLUSION

In this paper, we described the problem of storing ranges in TCAM, and presented a new multi-field range encoding scheme. In order to decrease the TCAM memory usage, we need to determine the codewords for all elementary regions. Based on the properties of hypercubes, finding the codewords for each elementary region become a feasible process. Although additional SRAM to store codewords is needed, it is fewer than other single-field schemes because our proposed schemes can use shorter codeword length. Compared with existing single-field encoding schemes, our proposed scheme uses 12% ~ 33% of TCAM memory needed in DRIPE or SRGE and 56% ~ 86% of TCAM memory needed in PPC for the classifiers of up to 10k rules. In order to further understand of our approach, we provide a detailed algorithm and experiment results in technical report [13].

REFERENCE

[1] A. Bremler-Barr and D. Hendler, "Space-Efficient TCAM-based Classification Using Gray Coding," in *IEEE INFOCOM*, 2007.

[2] A. Bremler-Barr, D. Hayy, and D. Hendler, "Layered Interval Codes for TCAM-based Classification," in *IEEE INFOCOM*, 2009.

[3] A. L. Buchsbaum, G. S. Fowler, B. Krishnamurthy, K.-P. Vo, and J. Wang, "Fast Prefix Matching of Bounded Strings", *ACM Journal of Experimental Algorithmics*, vol. 8, Jan. 2003.

[4] H. J. Chao, "Next Generation Routers," in *Proc. of the IEEE*, vol. 90, no. 9, pp.1518-1558, Sep. 2002.

[5] Y.-K. Chang and Y.-C. Lin, "Dynamic Segment Trees for Ranges and Prefixes," *IEEE Trans. Computers*, vol. 56, no. 6, pp. 769-784, June 2007.

[6] Y.-K. Chang and C.C. Su, "Efficient TCAM Encoding Schemes for Packet Classification using Gray Code," in *IEEE Globecom*, 2007.

[7] H. Che, Z. Wang, K. Zheng, and B. Liu, "DRES: Dynamic Range Encoding Scheme for TCAM Coprocessors," *IEEE Trans. on Computers*, vol. 57, no. 7, pp.902-915, June 2008.

[8] R. Cohen and D. Raz, "Simple Efficient TCAM Based Range Classification," in *IEEE INFOCOM Mini-Conference*, 2010.

[9] Q. Dong , S. Banerjee , J. Wang , D. Agrawal , A. Shukla, "Packet Classifiers in Ternary CAMs Can Be Smaller," in *ACM SIGMETRICS*, 2006,.

[10] F. Gray, "Pulse Code Communication," U. S. Patent 2 632 058, March 17, 1953.

[11] P. Gupta and N. McKeown, "Algorithms for Packet Classification," *IEEE Network*, vol. 15, no. 2, pp. 24–32, 2001.

[12] K. Lakshminarayanan, S. Venkatachary, and A. Rangarajan,"Algorithms for Advanced Packet Classification With Ternary CAMs," in *ACM SIGCOMM*, 2005.

[13] Y.-K. Chang, C.-I Lee and C.-C. Su, "Multi-Dimensional Range Encoding for Packet Classification in TCAM," NCKU *Technical Report*, http://cial.csie.ncku.edu.tw/publication/ncku-cial-2010-01.pdf

[14] H. Liu, "Efficient Mapping of Range Classifier into Ternary-CAM," in *IEEE HOTI*, 2002.

[15] J. Lunteren and T. Engbersen, "Fast and Scalable Packet Classification," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 4, pp.560-571, May 2003.

[16] A.X. Liu, C.R. Meiners, and E. Torng, "TCAM Razor: A Systematic Approach Towards Minimizing Packet Classifiers in TCAMs," *IEEE/ACM Trans. on Networking*, vol. 18, no. 2, pp. 490-500, Apr. 2010.

[17] O. Rottenstreich and I. Keslassy, "Worst-Case TCAM Rule Expansion," in *IEEE INFOCOM Mini-Conference*, 2010.

[18] O. Rottenstreich and I. Keslassy, "On the Code Length of TCAM Coding Schemes," in *IEEE ISIT*, 2010.

[19] D. Taylor, "Survey and taxonomy of packet classification techniques," *ACM Computer Surveys*, vol. 37, no. 3, pp. 238-275, Sep. 2005.

[20] D. Taylor and J. Turner, "ClassBench: A Packet Classification Benchmark," in *IEEE INFOCOM*, 2005.