ISBN 978-981-14-4787-7
Proceedings of 2020 the 10th International Workshop on Computer Science and Engineering (WCSE 2020)
Shanghai, 19-21 June, 2020, pp. 353-358
doi: 10.18178/wcse.2020.06.053

A Modified External Encoding Scheme for TCAM

Xinwei Liu [1], Zixin Guan [2+], Ting Huang [1] and Hui Li [1]

[1] Peking University Shenzhen Graduate School, Shenzhen 518055, China;

[2] Guangdong Communications & Networks Institute, Guangzhou 510670, China

Abstract. OpenFlow, one of the key technologies of Software Defined Network (SDN), has been applied widely in several years and its switches generally use TCAM (Ternary Content Addressable Memory) to achieve line-speed searching in flow table. As a parallel searching hardware, TCAM can provide very fast search but has some disadvantages including high cost, high power consumption and failing to support representing ranges. This paper proposes a modified external encoding scheme to solve the problem of represent ranges in TCAM, which searches an optimum boundary in external encoding and combines 0-1 bits counting to achieve the goal of reducing the complexity of algorithm and enhancing the performance of encoding. The experiments result shows that the proposed scheme can reach efficient encoding and reduce space consumption.

Keywords: OpenFlow; packet classification; TCAM; range encoding

1. Introduction

SDN, as a hot topic in network research, proposes to decouple the control and data planes. The forwarding of data plane uses OpenFlow protocol in SDN, and the lookup of flow table in OpenFlow is essentially a problem of packet classification. The packet classification is to compare the header information of incoming packet with the existing rules, and then perform the action of the matching rule with highest priority or drop the packet when there is no matching rule. Package classification algorithms can be divided into two categories [1, 2]. One is the software algorithms that search according to the data structure, like decision trees [3~11], have the advantages of high flexibility and expansibility but use more time during the lookup. The other is hardware algorithms that use TCAM to get higher searching speed but have shortcomings of high cost, high power consumption and failing to support representing ranges [12~25].

The representation of a range in TCAM is to encode the range into multiple triple entries consisting of three characters, "0", "1" and "*". The "*" in an entry is the wildcard character, which can represent either "0" or "1". For example, the range **[4, 7], [100,111]** in binary, can be encoded to a triple entry “1**” which means it can cover the number “100”, “101”, “110”, and “111” in binary. Internal encoding [12], external encoding [13] and SplitCoding [14] are some encoding methods proposed to solve the problem of range representation in TCAM. Nevertheless, the current encoding methods are inefficient and generate lots of entries, which leads to the problems of low space utilization and increased power consumption in TCAM.

This paper focuses on reducing the encoding number of TCAM entries to realize high space utilization by modifying external encoding. Because of the flexibility of range boundary in external encoding and the rule that the packet will be dropped when there is no matching entry in TCAM, the algorithm can find an optimum boundary of external encoding in order to result less encoding entries. Experimental result shows that the modified external encoding proposed in this paper can effectively solve the problem of encoding expansion in TCAM and has low complexity.

[+] Corresponding author.
E-mail address: guanzixin@gdcni.cn

2. Related Work

The first TCAM range encoding algorithm, internal encoding, was proposed by G. Varghese’s team in 1998[12]. Internal encoding can generate at most 2W-2 triple entries where W is the bit width and cause large space consumption.

In addition, the independence of ranges can be used to fix the problem of range representation in TCAM. Prof. H. Liu of Stanford University proposed an encoding algorithm [15] that encodes independent range in TCAM. The algorithm divides range that needs to be encoded into several independent small ranges and number these ranges. Furthermore, spare space in TCAM can be used to record these numbers so that we can locate a specific range quickly. The algorithm can reduce space consumption by only using spare space to record the numbers but updating is difficult when new rule is inserted, thereby the search efficiency will be reduced.

Prof. Ori Rottenstreich et al. of Israel Institute of Technology proposed the external encoding algorithm [13] to solve the problem of range expansion in 2013. External encoding mainly uses the sequence sensitivity of TCAM and is optimized on the basis of the internal encoding. External encoding mainly encodes the complementary set of the encode range. The result of external encoding is formed of the encoding entries of the complementary set on the top with the action DENY and the encoding entries of the universe set with ACCEPT on the bottom. The final result is obtained by comparing the entries number of internal encoding and external encoding. For range **[0, 6]**, the internal encoding would generate three triple entries “0**→ACCEPT”, “10*→ACCEPT”, and “110→ACCEPT”, while external encoding would generate two triple entries “111→DENY”, “***->ACCEPT”. External encoding reduce the range expansion upper bound from 2W-1 to W, which makes a great progress. After that Pro. Ori proposed optimal encoding [16] which gives the optimal number of encoding entries, and proves its range expansion upper bound. However, optimal encoding is complicated and difficult to apply in practice.

In 2019, SplitCoding [14] was proposed as a practical encoding scheme that combines internal encoding and external encoding. It firstly wipes off the longest common prefix of the two endpoints of the encoding range because the bits two endpoints shared do not need to be encoded. And then SplitCoding splits the range into two parts by the centre point of the two endpoints and compares the encoding results of internal encoding and external encoding of every part. Besides, SplitCoding uses 0-1 bit counting [17] to reduce the encoding complexity. Although SplitCoding generates less entries than internal encoding or external encoding does and is more practical, the range expansion need to be optimized.

3. Terms and Lemmas

This section will give some terms and lemmas for the description and readability in the follow sections.

3.1. Terms Definition

Longest Common Prefix (LCP): For a W-bit range R_{ab} = **[*a*, *b*]**, LCP is constructed by two parts, the common bits between *a* and *b* counting from high-order and the padding “*” in order to make LCP a W-bit triple entry.

General Extremal Range: For a W-bit range R_{ab} = **[*a*, *b*]**, the R_{ab} is called general extremal range if $a = 0$ or $b = 2^W$-1.

3.2. Related Lemmas

Lemma 1 [24]: For the general extremal range R = **[0, *b*]**, the least triple entries to cover the range R is the number of “1” in the binary representation of *b*+1. For every “1” bit in the binary representation of *b*+1, the triple entry is made by changing the “1” bit to “0” bit and filling the bits after the modified “0” bit with “*”. The encoding set is made of these triple entries. For example, a general extremal range R = **[0, 6]** can be encoded into three triple entries at least because of the “1” bit in the binary representation of 7(111) is 3. The encoding set includes “0**”, “10*” and “110”. Furthermore, if *b* and 0 have a common LCP, we should wipe off the LCP in *b* and use Lemma 1 in the rest of *b*.

Lemma 2 [24]: For a W-bit general extremal range R = **[*a*, 2^W-1]**, the least triple entries to cover the range R is the number of “0” in the binary representation of *a*-1. For every “0” bit in the binary representation of *a*-

1, the triple entry is made by changing the “0” bit to “1” bit and filling the bits after the modified “1” bit with “*”. The encoding set is made of these triple entries. For example, a general extremal range R = **[4, 7]** can be encoded into one triple entry at least because of the “0” bit in the binary representation of 3(011) is 1. The encoding set includes “1**”. Furthermore, if a and 2^W-1 have a common LCP, we should wipe off the LCP in a and use Lemma 2 in the rest of a.

4. Modified External Encoding

In this section, we will show some observations of external encoding and introduce our design in detail. An encoding example is given as well.

4.1. Observations

The external encoding does not encode the target range set S but encode the complement set S’ and universe set in W bits in order to generate less TCAM entries. However, the boundary of the complement set that external encoding choose might too large, like W-bit width range **[0, *a*]** use 2^W-1 as the external encoding boundary, which may result in generating more TCAM entries.

Besides, the packet that does not match any rules in TCAM will be dropped. The main idea of the proposal algorithm is that we could narrow the boundary of external encoding and encode less area. Table 1 shows the encoding results of internal encoding, external encoding and modified external encoding using **[0, 78]** as an example. The modified external encoding narrows the external encoding boundary and results less TCAM entries.

Table 1: Example of encoding scheme

Encoding range	[0,78]=[0000000,1001110]→ACCEPT	
Internal Encoding	0******→ACCEPT 1000***→ACCEPT 10010**→ACCEPT 100110*→ACCEPT 1001110→ACCEPT	5 entries

Encoding range	[79,127]=[1001111,1111111]→DENY [0,127]=[0000000,1111111]→ACCEPT	
External Encoding	11*****→DENY 101****→DENY 1001111→DENY *******→ACCEPT	4 entries

Encoding range	[79,79]=[1001111,1001111]→DENY [0,79]=[0000000,1001111]→ACCEPT	
Modified External Encoding	1001111→DENY 0******→ACCEPT 100****→ACCPET	3 entries

4.2. Proposed Method

Base on the observation above, we propose a modified external encoding. For a W bits range **[0, *a*]**, we set x equals the number of “0” in binary representation of a. For every “0” bit in binary representation of a, we set all the bits after every “0” to “1” to build different boundary B_i (1<=i<=x). Apparently, different B_i has different LCP with a, the algorithm is to find the minimum value of encoding number summary of **[0, B_i]** and **[a+1, B_i]**. The number of entries that modified external encoding generates is (1).

$$\text{Number of entries} \ = \ \text{Min}\{1 \leq i \leq x | L1([0,\ B_i]) + L2([a+1,\ B_i])\} \qquad (1)$$

L1 is the operation of Lemma 1 and L2 is the operation of Lemma 2. Using lemmas in [24] that allows us to get a simple way to encode. The external encoding actually is included in the modified external encoding, so the final result is the comparison of modified external encoding and internal encoding.

For example, **[0, 78]** in bit width W=7, the binary representation is **[0000000, 1001110]**. According to Lemma 1, the number of entries using internal encoding is 5 since the number of “1” bit in binary representation of 79(1001111) is 5. The number of entries using external encoding is 4 since the number of “0” bit in binary representation of 79(1001111) is 3 according to Lemma 2 and the number of encoding entry of **[0000000, 1111111]** is 1. The number of entries using modified external encoding is:

Number of entries = Min {{L1 ([0000000, 1001111]) +L2 ([1001111, 1001111])},

{L1 ([0000000, 1011111]) +L2 ([1001111, 1011111])},

{L1 ([0000000, 1111111]) +L2 ([1001111, 1111111])}}

=min {2+1, 2+2, 1+3}

= 3

Compare to internal encoding, the final encoding entries of **[0, 78]** are "1001111→Deny", "0******→Accept", and "100****→Accept".

Modified external encoding traverses the bits in *a* to construct different LCP to find the optimum external boundary by flipping "0" bit to "1" bit. Thanks to the lemmas mentioned before, modified external encoding can reach the goal of high encoding speed and high space utilization.

Similarly, the encoding of extreme range **[*b*, 2^W-1]** is to construct different external encoding boundary by flipping the "1" bit into "0" in binary representation of *b*

5. Evaluation

The simulation of this experiment is written in C++ and the operating environment is Intel(R) Xeon(R) Silver 4114 CPU @2.20 GHz CPU, 66G memory in Ubuntu 16.04. The experiment is divided into two parts one is the average range expansion ratio of different width and one is range expansion distribution over all possible ranges when width is 16.

5.1. Average Range Expansion Ratio

For different bit width W, the average range expansion ratio is the average encoding number of a range in a corresponding width. Such as W=2, the ranges need to be encoded are **[0, 0]**, **[0, 1]**, **[0, 2]**, **[0, 3]**, **[1, 1]**, **[1, 2]**, **[1, 3]**, **[2, 2]**, **[2, 3]**, **[3, 3]** and we should evaluate the average encoding number of these 10 ranges. Fig. 1 shows the average range expansion ratio of internal encoding, external encoding and proposed method. The proposed method can result fewer entries in average compared with internal encoding, external encoding and SplitCoding, which means the proposed method can reduce space and decrease the power consumption in TCAM.

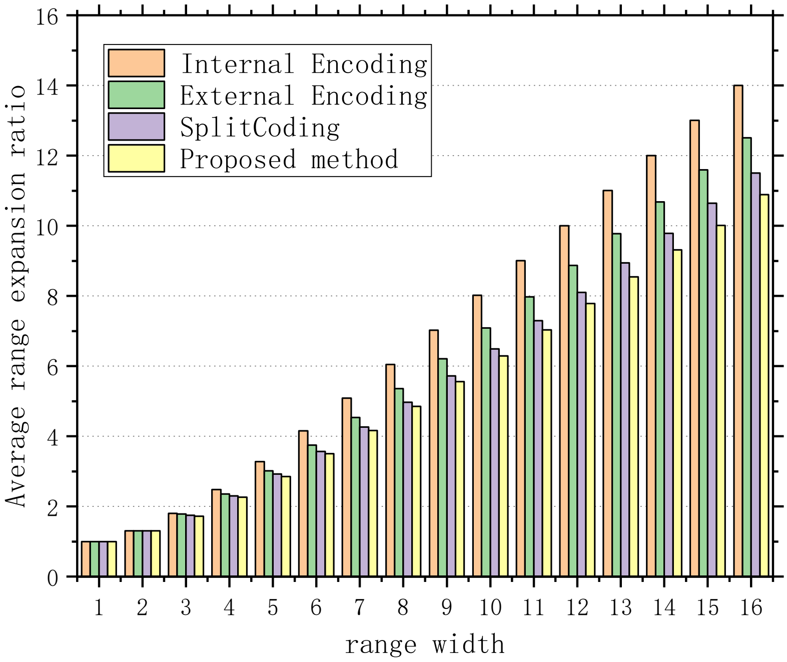

Fig. 1: Average range expansion ratio for diffierent width

5.2. Range Expansion Distribution

Range expansion distribution shows the distribution of the encoding expansion ratio for a given width W = 16. Fig.2 shows the range expansion distributions of external encoding and proposed method. The range expansion distribution of proposed method is on the most left in Fig. 1, which means the proportion of small expansion is larger in proposed method than that in external encoding and SplitCoding. The result shows that proposed method generates less entries than external encoding and will use less TCAM space.

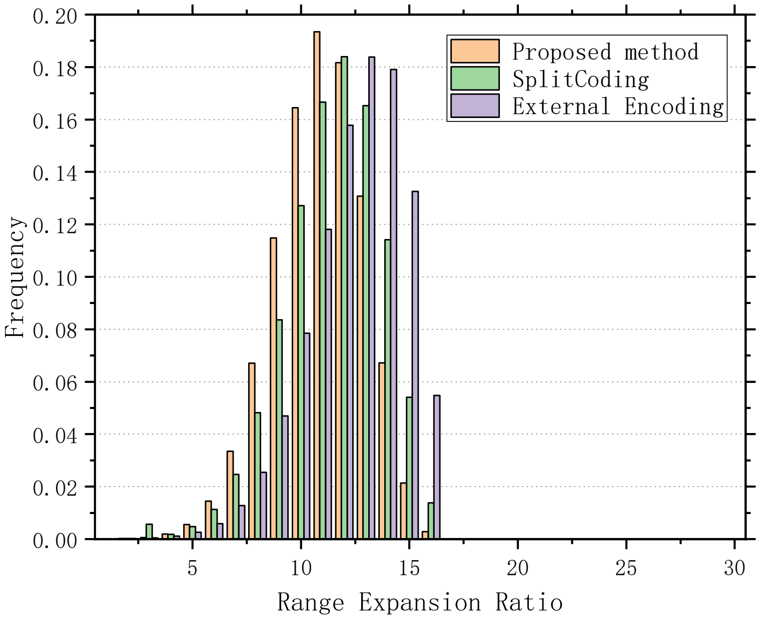

Fig. 2: Range expansion distribution over all possible ranges for W = 16

6. Conclusion

In this paper, a modified external scheme is proposed to solve the problem of TCAM range encoding expansion. Taking advantage of the rule of dropping when there is no matches in TCAM, the proposed algorithm can find an optimum boundary of external encoding in order to encode less entries for a range. Experimental result shows that the proposed scheme can effectively reduce the problem of encoding expansion, reduce storage consumption and improve encoding performance.

7. References

[1] Taylor D E. Survey and taxonomy of packet classification techniques [J]. ACM Computing Surveys (CSUR), 2005, 37(3): 238-275.

[2] Qi Yaxuan, Li Jun. Theoretical analysis and algorithm design of high-performance packet classification algorithms [J]. Chinese Journal of Computers, 2013, 36(02): 408-421.

[3] Gupta P, Mckeown N. Packet classification using hierarchical intelligent cuttings [C]// Proc of IEEE Hot Interconnects. Piscataway, NJ: IEEE Press, 1999: 34-41.

[4] Singh S, Baboescu F, Varghese G, et al. Packet classification using multidimensional cutting [C]// Proc of ACM SIGCOMM. ACM Press, 2003: 213-224.

[5] Qi Yaxuan, Xu Lianghong , Yang Baohua, et al. Packet classification algorithms: from theory to practice [C]// Proc of IEEE INFOCOM. Piscataway, NJ: IEEE Press, 2009: 648-656.

[6] Vamanan B, Voskuilen G, Vijaykumar T N. EffiCuts: optimizing packet classification for memory and throughput [C]// Proc of ACM SIGCOMM. ACM Press, 2010: 207-218.

[7] Li Wenjun, Li Xianfeng. HybridCuts: a scheme combining decomposition and cutting for packet classification [C]// Proc of IEEE Hot Interconnects. Piscataway, NJ: IEEE Press, 2013: 41-48.

[8] He Peng, Xie Gaogang, Salamatian. K, et al. Meta-algorithms for software-based packet classification [C]// Proc of IEEE ICNP. Piscataway, NJ: IEEE Press, 2014: 308-319.

[9] Li Wenjun, Li Xianfeng, Li Hui, et al. CutSplit: a decision-tree combining cutting and splitting for scalable packet classification [C]// Proc of IEEE INFOCOM. Piscataway, NJ: IEEE Press, 2018: 2645-2653.

[10] Liang E, Zhu H, Jin X, et al. Neural packet classification [C]// Proc of ACM SIGCOMM. ACM Press, 2019: 256-269.

[11] Li Wenjun, Yang Tong, Chang Y-K, et al. TabTree: a tss-assisted bit-selecting tree scheme for packet classification with balanced rule mapping [C]// Proc of ACM/IEEE ANCS. Piscataway, NJ: IEEE Press, 2019 .

[12] Srinivasan. V, Varghese. G, Suri. S, et al. Fast and scalable layer four switching [C]// Proc of ACM SIGCOMM. ACM Press, 1998: 191-202.

[13] Rottenstreich O, Cohen R, Raz D, et al. Exact worst-case TCAM rule expansion [J]. IEEE Trans on Computers, 2013, 62(6): 1127-1140.

[14] Li W, Li D, Liu X, et al. A power-saving pre-classifier for TCAM-based IP lookup [J]. Computer Networks, 2019, 164: 106898.

[15] Li Wenjun, Li Xianfeng, Li Hui. MEET-IP: memory and energy efficient tcam-based ip lookup [C]// Proc of IEEE ICCCN. Piscataway, NJ: IEEE Press, 2017: 1-8.

[16] Rottenstreich O, Keslassy I, Hassidim A, et al. Exact worst-case TCAM rule expansion, Optimal in/out TCAM encodings of ranges [J]. IEEE/ACM Transactions on Networking, 2016, 24(1): 555-568.

[17] Schieber B, Geist D, Zaks A. Computing the minimum DNF representation of Boolean functions defined by ranges[J]. Discrete Applied Mathematics, 2005, 149(1-3): 154-173.

[18] Sun Penghao, Lan Julong, Lu Xiaoyuan, et al. Field-trimming compression model for rule set of packet classification. Journal of Electronics & Information Technology, 2017, 39(05): 1185-1192.

[19] Sun Penghao, Lan Julong, Wang Peng, et al. RFC: range feature code for TCAM-based packet classification [J]. Computer Networks, 2017, 118: 54-61.

[20] Li Wenjun, Liu Xinwei, Le Wenxia, et al. A practical range encoding scheme for TCAMs [C]// Proc of ACM SIGCOMM Posters and Demos. ACM Press, 2019: 163-165.

[21] Li Wenjun, Li Dagang, Liu Xinwei, et al. A power-saving pre-classifier for TCAM-based IP lookup[J]. Computer Networks, 2019, 164: 106898.

[22] Li Xianfeng, Lin Yuanxin, Li Wenjun. GreenTCAM: a memory- and energy-efficient TCAM-based packet classification [C]// Proc of IEEE ICNC. Piscataway, NJ: IEEE Press, 2016: 1-6.

[23] Liu Huan. Efficient mapping of range classifier into ternary-cam [C]// Proc of IEEE Hot Interconnects. Piscataway, NJ: IEEE Press, 2002: 95-100.

[24] Lakshminarayanan K, Rangarajan A, Venkatachary S. Algorithms for advanced packet classification with ternary cams [C]// Proc of ACM SIGCOMM. ACM Press, 2005: 193-204.

[25] Bremler-B A, Hendler D. Space-efficient TCAM-based classification using gray coding [C]// Proc of IEEE INFOCOM. Piscataway, NJ: IEEE Press, 2007:1388-1396.