Computer Networks 118 (2017) 54–61

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

RFC: Range feature code for TCAM-based packet classification

Penghao Sun*, Julong Lan, Peng Wang, Teng Ma

China National Digital Switching System Engineering & Technological R&D Center, Zheng Zhou, 450001, China

ARTICLE INFO

Article history:
Received 21 June 2016
Revised 11 November 2016
Accepted 26 February 2017
Available online 28 February 2017

Keywords:
TCAM
Packet classification
Range expansion

ABSTRACT

Packet classification has been widely used in various Internet applications, including the recent hot topic SDN. Generally, TCAM is a typical device for high-speed packet classification. However, when it comes to some classification rules that contain ranges, since TCAM is not well designed to represent ranges, range expansion problem is caused. Range expansion could cost much more TCAM entries than the number of rules, thus impairing the utilization of TCAM. In practice, there are some unused bits in a TCAM entry, which could be used to reduce the range expansion. In this paper, we propose a scheme to efficiently represent ranges with such extra bits. Our scheme is based on the observation that in prior encoding schemes that use extra bits to represent ranges, the encoding in extra bits and its fallback scheme are always regarded as two totally separate processes. Even though the two encoding methods are different, as they both focus on the same rule, there actually exists some relevance in information of the two encoding process. In our scheme, the two encoding processes are brought together with range feature code (RFC) as the link, thus reducing the information redundancy between them. Experiment results show that when 36 extra bits are available in TCAM, our scheme reduces the redundancy of range rules by around 36% compared with the best prior scheme.

1. Introduction

Packet classification plays an important role in various Internet devices and applications. Especially, the recently arisen hot topic SDN [1] also relies heavily on packet classification to perform its flexibility in dataplane. The key point in packet classification is to compare the header field of every incoming packet to a group of preset rules, and then pick up one rule that matches the packet with the highest priority. To keep the packet being processed in line speed and the network running in appropriate scale, packet classification is often required of a line speed processing and an adequate rule capacity.

There are mainly two types of rules in the matching process of packet classification, one being exact match and the other wildcard match. For packet filtering, load balancing and many other applications, wildcard match is widely used, which means that one rule could be matched by more than one packet header with certain bits assigned to "*" (which means "don't care" in this bit) instead of an exact number "0" or "1". To guarantee the lookup speed of wildcard matching, ternary content-addressable memory (TCAM) becomes a popular choice for such function. In TCAM, each entry consists of a fixed number of ternary digits, which could be assigned with a value of 0, 1 or *. When a packet comes, parallel look-up process is enabled in a TCAM, thus ensuring the line speed of matching process. Generally, when there are more than one entry matched, TCAM returns the index of the first one as the output.

Efficient as TCAMs are, their scalable usage is greatly restrained in packet classification. For range representation, TCAMs are also not well suited. An exact cover of a certain range such as port number requires often more than one TCAM entry, which is called range expansion problem. On the other side, both the high hardware cost and high power consumption restrict greatly the expansion of TCAM storage capacity (Mohan and coworkers [2,8] did researches on the improvement of architecture in TCAM). Therefore, reduction of range expansion in TCAM is of great importance.

To alleviate the storage pressure of TCAM, there are many researches on the reduction of range expansion. Typically, there are several different methods, such as improvement of encoding scheme, hardware modification, and even multi-TCAM based lookup. In this paper, we concentrate on the encoding of extra bits in TCAM to alleviate the range expansion problem. Based on our observation, each range has a feature code that remains constant among all code values in this range. For the same code in extra bits, range feature code can help differentiate different ranges in the fallback code. Therefore, different ranges could share the same code in extra bits. With a good exploitation of range feature code, the utilization of extra bits in TCAM can get an obvious improvement. More precisely, our the contributions are as follows: (1) we identify the redundancy in the encoding schemes

* Corresponding author.
E-mail address: sphshine@126.com (P. Sun).

http://dx.doi.org/10.1016/j.comnet.2017.02.016
1389-1286/

where encoding of extra bits and encoding of basic scheme are considered as two non-relevant stages; (2) we present a series of algorithms that couples the encoding in fallback codes and extra bits in a single decision stage, which takes advantage of the orthogonality in range feature code to improve the utilization of extra bits; (3) we propose a fallback encoding scheme that both maintains the advantage of range feature code and reduces the range expansion problem introduced by short ranges.

2. Related work

Compression of the storage space in TCAM has attracted a lot of attention in the past few years. Since the size of TCAM is limited and the usage of TCAM brings much convenience to the matching process of packet header which contains prefix, wildcard or range, the improvement of TCAM utilization is always a hot topic in both academia and industries. For example, Liu and coworkers [3–7,15] did some researches on the algorithm scope in compression, while Taylor [9] tried to solve this problem with the combination of taxonomy preprocessing. Also, multi-TCAM based schemes are studied in [21–23], which improves the lookup speed and updating speed.

In this section, a brief description of the prior art of optimizing range representation in TCAM is given, which is also the main point of this paper. Basically, we put the researches into two categories: database-independent range encoding and database-dependent range encoding.

2.1. Database-independent range encoding

Prefix expansion of ranges is one well-known database-dependent range encoding scheme in TCAM. Taylor [9] gives us an illustration of how a range could be represented with a set of prefixes in TCAM. However, this method may cost too many entries in TCAM in certain cases. For example, in a rule that contains more than one range fields, each rule would be expanded into several different prefixes, and the overall number for the representation of this rule is the cross product of all entry number of separate range representations. It's proven that this encoding method has a worst-case expansion ratio of 2W-2 for a *W*-bit range. Also, a case in [9] shows that the worst-case representation for a 16-bit range requires 30 prefixes. What's more, things could even get worse for a typical IP ACL rule which contains two range fields (source port number and destination port number), whose worst-case expansion is $30^2 = 900$ entries.

Other database-independent range encoding typically uses techniques of modifying the code format in TCAM. With a carefully designed code format, the ranges in TCAM could be represented with less entries, thus reduce the overall expansion ratio. Lakshminarayana et al. [5] proposed a novel algorithm called DIPRE whose idea derived from fence coding. With the idea of fence encoding, DIPRE separate the string of a code into several sub-strings. In each sub-string, fence encoding is used to replace original binary code. Such scheme requires more bits to encode an integer while making the code of continuous integer closer to each other, thus obtaining a better performance in the reduction of range expansion. In DIPRE, the more extra bits are used for encoding, the better compression ratio this scheme could achieve. Biemer and Hendler [10] proposed another efficient encoding scheme called SRGE. SRGE encode the integer according to binary reflected tree of GRAY code. With the reflection property of binary reflected tree and the adjacent property of GRAY code, SRGE makes sure that any range could be compressed with one or more folds of the nodes in the binary reflected tree. In contrast to DIPRE, SRGE achieves a stable compression performance with fixed length of code. Bit Weaver [14] is also a great tool that could be combined with other encoding scheme to further compress entries stored in TCAM.

2.2. Database-dependent range encoding

Database-dependent encoding usually employs some modifications in hardware to help carry out the encoding process. Hierarchical database-dependent encoding schemes are proposed in [3,4]. In these schemes, ranges are divided into different regions, and the encoding process contains both the encoding of regions and encoding of ranges within each region. However, these schemes are not very effective in the reduction of expansion. Chang et al. [13] regards the field value of one or more range fields as a multi-dimensional space. By analyzing the range value distribution, the space could be divided into some cyber blocks. Orange [25] design a double pipeline of TCAM for the match process of a range. In this scheme, the space defined by a range field is split by the lower bound and upper bound of a range. Two TCAM pipelines correspond respectively to the lower bound and upper bound of ranges. Rottenstreich et al. [26] redefined the usage of TCAM, and use "in" and "out" instructions in TCAM for a range. This innovative work needs a modification of the architecture of TCAM. In some schemes, surplus bits in TCAM are taken advantage of to encode some ranges that cause big expansion and appear frequently in a database. DIPRE [5] and SRGE [10] also introduces the idea of database-dependent range encoding to further reduce the expansion ratio, but the encoding schemes for extra bit does not have an efficient utilization, in which every range cost an entire bit position. DRES [11] proposed a range selection algorithm to encode ranges with extra bits which is based on greedy algorithm. LIC [12] made a further analysis of ranges with large expansion and proposed a range representation scheme with efficient extra bits utilization ratio in TCAM. They presented several versions of the scheme and achieved a significant improvement in the utilization of extra bits. However, they separate completely the encoding process of extra bits and their fallback scheme, ignoring that they are closely related to each other. Chang et al. [13] proposed a multi-field range encoding scheme which is based on SRAM for pre-lookup. This scheme eliminated range expansion problem, but relies heavily on much SRAM space.

3. Range feature code based encoding scheme

3.1. Basic idea of range feature code

Utilization of extra bits in TCAM is a popular method to further reduce the expansion caused by large ranges. However, observation of previous works show that all the schemes proposed in the past treated the basic encoding scheme and the encoding of extra bits as two totally separate problems. For example, when considering the encoding process in extra bits, [5,10,12] all omit the feature of the basic encoding scheme (known as fallback scheme). Since encoding of extra bits and basic encoding scheme aim at the same rule, such previous methods just neglected the relation between them, thus introducing some information redundancy in the encoding process. Here we introduce the range feature code encoding scheme that systematically deals with the encoding in TCAM.

Definition 1. Range Feature Code. Range feature code is a group of digits that remains constant among all the code values in a range, and other digits are set to "*"in a range feature code. Specifically, we use Rfc(r) to denote the range feature code of range r.

For example, in a 16-bit port number field which is represented with prefixes, the highest 6 digits remain constant as the value "000000″ in the range [0, 1023]. Therefore, the 16-bit string "000000**********" is the range feature code of range [0, 1023]. Range feature code partly reflects the feature of a range in a certain encoding scheme, just as its name shows.