一种基于匹配域裁剪的包分类规则集压缩方法

孙鹏浩*① 兰巨龙① 陆肖元② 胡宇翔① 马 腾①
①(国家数字交换系统工程技术研究中心 郑州 450002)
②(上海未来宽带技术及应用工程研究中心 上海 200336)

摘 要：随着以 OpenFlow 为代表的多匹配域包分类规则的出现，匹配域数量的不断增加、流表宽度的不断增大以及流表规模的不断膨胀，大大增加了硬件存储的压力。为提高现有三态内容可寻址存储器(TCAM)资源利用率，该文提出一种基于规则集特征分析的匹配域裁剪模型 Field Trimmer。一方面基于对规则集中匹配域的逻辑关系分析，实现匹配域的合并，从而减少匹配域的数量；另一方面基于对规则集统计规律的分析，实现匹配域的裁剪，使用部分匹配域来达到整体的匹配效果。实验结果表明，相比于其他方案，该方案在较小的时间复杂度下，能够进一步节省 OpenFlow 流表的 TCAM 存储空间需求 50%左右；对于常见的包分类规则集，该方案所需的储存空间能够节省 40%以上。

关键词：包分类；三态内容可寻址存储器；OpenFlow
中图分类号：TP393 **文献标识码**：A **文章编号**：1009-5896(2017)05-1185-08
DOI: 10.11999/JEIT160740

Field-trimming Compression Model for Rule Set of Packet Classification

SUN Penghao① LAN Julong① LU Xiaoyuan② HU Yuxiang① MA Teng①
①(*National Digital Switching System Engineering & Technology Research Center*, *Zhengzhou* 450002, *China*)
②(*Shanghai Engineering Research Center for Broadband Technologies & Applications*, *Shanghai* 200336, *China*)

Abstract: With the emergence of multi-field packet classification such as OpenFlow, the increasing number of match fields, continuous growth in bit-width of entries and ever growing scale of rule set all bring much pressure on the storage space in hardware. To improve the utilization of the existing Ternary Content Addressable Memory (TCAM) resources, a match field reduction scheme Field Trimmer is proposed based on the analysis of rule feature. On the one hand, with the analysis of logical relationships among different match fields, some fields can be merged to reduce the number of match fields. On the other hand, with the analysis of statistical features in a rule set, some of the match fields are picked up to achieve the classification function of the whole set. Experiment result shows that with less algorithm complexity, the proposed scheme can save around 50% storage space in the rule set of OpenFlow compared to the best prior art, and about 40% storage space in the popular 5-tuple packet classification rule set.

Key words: Packet classification; Ternary Content Addressable Memory (TCAM); OpenFlow

1 引言

当前网络功能中，路由选择、防火墙、访问控制列表、流量调度等技术都依赖于包分类技术的实现。随着当前软件定义网络[1](Software-Defined Networking, SDN)等新技术的兴起，上层应用对于数据平面的数据细粒度管控需求进一步提升，使包分类的规则集规模不断增大，维度不断增加。例如 OpenFlow[2]就是通过增加匹配域的数量来实现对数据平面数据流细粒度的管控(OpenFlow v1.3 规定了 40 个匹配域字段，并规定交换机必须支持其中的 13 个匹配域)。

目前，基于包分类技术对于匹配速度的要求、对前缀匹配和范围匹配的需求(例如 IP 地址的匹配和端口号范围的匹配)，三态内容可寻址存储器(Ternary Content Addressable Memory, TCAM)被广泛应用于包分类技术的实现上。然而，由于 TCAM 功耗大、成本高，其存储规模受到了极大的限制；同时，规则集的储存方式不当，也会明显增加对 TCAM 表项的占用，例如，对于端口号范围的储存，最坏情况下一个范围需要 30 条 TCAM 表项，而考虑源端口号和目的端口号共存的情况下，最坏

收稿日期：2016-07-11；改回日期：2017-02-08；网络出版：2017-03-20
*通信作者：孙鹏浩 sphshine@126.com
基金项目：国家 973 计划项目(2012CB315901)，国家自然科学基金(61521003)，国家“863”计划项目(2013AA013505)
Foundation Items: The National 973 Program of China (2012CB315901), The National Natural Science Foundation of China (61521003), The National 863 Program of China (2013AA013505)

情况下将需要 900 条表项[3]， 这类问题称之为范围扩张问题。目前的包分类技术按照实现方法划分，主要有基于RAM的算法实现方案和基于TCAM的硬件实现方案。不同的实现模式主要涉及的优化问题包括：端口号范围扩张问题、表项合并问题和表项更新问题。

基于 RAM 的算法实现方案，主要通过多维空间切割、决策树优化和哈希等方法进行。在参考文献[4]的方案中，将整体的规则集视为一个多维空间，通过对多维空间以一定的算法进行划分，最终实现优化的规则表达方式，从而减少了存储表项，算法的复杂度较高。依赖决策树实现的查表方案主要围绕决策树节点的剪枝合并问题展开，其难点在于算法时间复杂度和空间复杂度的权衡问题。文献[5]提出了一种能够根据不同场景在时间复杂度和空间复杂度灵活切换的场景，但随着匹配域的增加，依赖于决策树实现的匹配算法很难真正解决空间复杂度问题。文献[6]提出了基于哈希算法的查表模式，能够在大部分情况下实现快速查表，但是难以在最坏情况下保证线速查表。

由于 TCAM 在包分类实现中的广泛应用，基于TCAM 的硬件实现方案存在着广泛的研究基础。对于范围扩张问题，在实际的编码方案上，DIPRE[3]和 SRGE[7]分别通过硬件预编码的方式，将端口号范围字段重新编码，使其编码格式在 TCAM 中更加适合合并，在一定程度上减少了范围扩张问题。LIC[8]设计了一种利用 TCAM 表项中多余 bit 来对扩张问题比较严重的端口号范围单独编码的实现方案，能够在很大程度上减少范围扩张问题。然而这种方案依赖于对规则库的预先分析并提取特征，并且对硬件支持具有较高的依赖性。文献[9,10]在对 TCAM 中不同表项的数学特征进行分析后，提出了基于数值特征的表项合并方案，在一定程度上减少了 TCAM 表项的使用量。文献[11–13]提出了基于 TCAM 的新型硬件匹配流水线结构，改变了单一TCAM 一次匹配的模式，以索引值传递的方式实现了 TCAM 的分级匹配查找，减少了多匹配域组合引起的表项扩张问题。然而，这类方案的实现基于协议树的分析实现，多匹配域下计算复杂度较高。文献[14]提出了基于多级分层的规则预压缩方案减少了 TCAM 的存储空间要求，但是实现方案转为复杂。文献[15]提出了基于特定硬件架构的范围匹配实现方案，通过两条 TCAM 流水线实现了端口号范围的匹配,缓解了范围扩张问题。文献[16,17]立足于对规则集的特征分析，提出了使用少数匹配域或数位完成匹配过程的方案，理论上能够实现较高的压缩效果，但计算复杂度过高，近似算法的性能偏离较大。文献[18]根据 OpenFlow 多域之间的逻辑关系将流表划分为子流表进行查表，但是压缩后对于硬件资源的节约效果并没有很理想。文献[19]设计了基于 FPGA 的 TCAM 与处理方案，需要硬件预处理较多。

基于上述分析，本文提出了以 OpenFlow 协议为代表的多匹配域规则集的压缩方案。通过对 OpenFlow 等规则集特征的分析，本文提出了匹配域合并裁剪模型 Field Trimmer，主要在以下几个方面做出了贡献：建立了以 OpenFlow 协议为例的匹配域逻辑关系分析模型，通过模型分析，实现匹配域合并，减少在匹配过程中冗余的匹配域信息，进而缩减了表项宽度；建立特定匹配域的裁剪算法，在不影响匹配结果的情况下，减少匹配域的使用，从而进一步减少 TCAM 资源的消耗。

2 模型描述

通过对 OpenFlow 支持的多种匹配域分析发现，许多匹配域之前存在着明确的逻辑关系，使匹配域之间具有强相关性(详细内容参考第 3.1 节)。这种特性的存在，使表项宽度所能确定的信息范围包含了很大的冗余。同时，由于实际的规则集规模有限，过多的匹配域对于数据包分类的功能实现上也存在一定的冗余(详细内容参考第 3.2 节)。本文主要从匹配域之间的逻辑关系和匹配域的统计特征两方面建立压缩模型。

现将本文中出现的主要变量定义描述如下：对于一个具有 N 条规则的包分类规则集 χ , $F_1,F_2,\cdots,F_m$ 表示其所包含的 m 个匹配域。χ 中的规则定义为 R，其中，第 i 条规则定义为 R_i $(1\le i\le N)$，而 R_i 中的第 j 个匹配域 F_j $(1\le j\le m)$取值定义为 $f_j(i)$。对于不同的匹配域，$f_j(i)$ 可以是精确数值、前缀(IP 地址)或者整数范围(端口号)。χ 中规则的位宽定义为 $W(\chi)$。

2.1 基于逻辑关系的匹配域合并

以 OpenFlow 为代表的多匹配域包分类规则由于单一表项宽度过大，影响了表项规模的可扩展性。OpenFlow v1.3 中所规定的交换机必须支持的 13 个匹配域中，不同的匹配域并不是完全独立取值。本文将不同匹配域之间的逻辑关系主要分为 3 类。

定义 1 对立关系：对于两个匹配域 F_i 和 F_j，若一个数据包的包头域不可能同时包含域 F_i 和域 F_j 的相应取值，则称域 F_i 和域 F_j 是对立关系。相应地，$\langle F_i,F_j\rangle$ 称为一个对立关系对 P_O，表示为 $\langle F_i,F_j\rangle\in P_O$。例如，TCP 源端口地址和 UDP 源端口地址

<TCP_SRC, UDP_SRC>构成一个对立关系对 P_O。

定义 2　依赖关系：对于两个匹配域 F_i 和 F_j，若一个数据包头中的 F_j 存在当且仅当 F_i 存在，则称域 F_j 对于域 F_i 是依赖关系。相应地，$\langle F_i, F_j \rangle$ 称为一个依赖关系对 P_C，表示为 $\langle F_i, F_j \rangle \in P_C$。例如，IPv6 的源地址对于 IPv6 的目的地址之间属于依赖关系。

定义 3　演进关系：对于两个匹配域 F_i 和 F_j，若 F_j 在协议内容上是 F_i 演进，则称域 F_i 和域 F_j 是演进关系。相应地，$\langle F_i, F_j \rangle$ 称为一个演进关系对 P_E。演进关系的典型代表是 IPv4 协议和 IPv6 协议。

对于 OpenFlow v1.3 中规定的交换机所必须支持的 13 个域(IN_PORT, ETH_DST, ETH_SRC, ETH_TYPE, IP_PROTO, IPV4_SRC, IPV4_DST, IPV6_SRC, IPV6_DST, TCP_SRC, TCP_DST, UDP_SRC, UDP_DST)，以上 3 种关系如关系矩阵 $\boldsymbol{\Pi}$ 所示(13 个匹配域在矩阵中的顺序如上所述)：

$$\boldsymbol{\Pi}=\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & P_C & P_C & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & P_C & 0 & P_C & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & P_C & P_C & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & P_C & P_E & P_O & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & P_C & 0 & P_O & P_E & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & P_E & P_O & 0 & P_C & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & P_O & P_E & P_C & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & P_C & P_O & P_O\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & P_C & 0 & P_O & P_O\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & P_O & P_O & 0 & P_C\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & P_O & P_O & P_C & 0
\end{bmatrix}$$

对于对立关系对 $\langle F_i, F_j \rangle \in P_O$，假设 F_i 的位宽为 l_i，F_j 的位宽为 l_j。交换机解析电路在解析数据包过程中，通过对协议的识别，能够区分数据包的 P_O 中实际存在的匹配域 F_i 或 F_j。将原本 l_i 位宽的 F_i 和 l_j 位宽的 F_j 合并为一个匹配域 $F_{i,j}$，合并后位宽为

$$l_{i,j} = \max(l_i, l_j) + \text{flag} \tag{1}$$

其中，增加的 flag 标志位是用以区分合并域的取值来源，对于典型的两个匹配域 $\langle F_i, F_j \rangle \in P_O$，只需 1 bit 即可。标志位由解析电路赋值，例如，0 代表合并域中实际值为 F_i，1 代表 F_j。对 P_O 中两个匹配域的合并，并不影响实际数据包匹配结果。对于一次 P_O 的匹配域合并，TCAM 中所需的表项宽度减少了 $(l_i + l_j - l_{i,j} - 1)$ bit。

对于演进关系对，典型代表是 IPv4 地址和 IPv6 地址。在 IPv6 协议[20]中，源地址和目的地址位宽由 IPv4 的 32 位扩展为 128 位，并且在相关的功能控制上，IPv6 也对 IPv4 进行了功能扩展。因此，IPv6 在功能上包含了 IPv4，在实际使用中，可以通过从 IPv4 向 IPv6 的转换，从而节省 IPv4 地址位宽。RFC 6052[21]中明确的 IPv4 向 IPv6 地址转换协议如图 1。

IPv4 协议向 IPv6 协议转换过程中，只需要考虑源地址和目的地址之间的转换。而通过选取特定的转换方案和在 IPv6 地址中的明确特定 IPv4 前缀后缀，可以避免引起地址冲突。

通过以上分析，可以发现，对立关系对 P_O 和演进关系对 P_E 的提取合并，可以实现基于逻辑关系的匹配域合并，能够使表项宽度缩减，并且减少了流表中匹配域的数量。匹配域的减少为进一步实现匹配域裁剪减少了计算复杂度，提高了匹配域裁剪的实现效率。

2.2 基于统计特征的匹配域裁剪

以图 2 中的规则集为例，图 2 中的规则集共有 $F_1 \sim F_3$ 3 个匹配域，由 $R_1 \sim R_4$ 4 条规则组成。对于

PL	0----------32----40----48--56---64---72---80---88---96--104----------
32	prefix | v4(32) | u | suffix
40	prefix | v4(24) | u | (8) | suffix
48	prefix | v4(16) | u | (16) | suffix
56	prefix | (8) | u | v4(24) | suffix
64	prefix | u | v4(32) | suffix
96	prefix | v4(32)

图 1　RFC 6052 提出的 IPv4 向 IPv6 地址转换方案

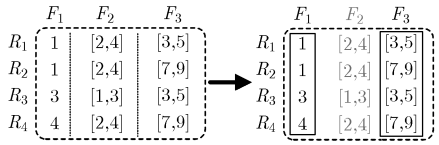

图 2 匹配域裁剪示意图

一个数据包头，匹配域 F_1 可确认包头是否符合 R_3 或 R_4，而再借助 F_3，即可进一步确认包头是否符合 R_1 或 R_2。因此，将匹配域 F_2 裁剪，仅靠匹配域 F_1 和 F_3 就可完成对整个规则集中 4 条规则的区分。例如，对于包头 $P=(1,3,4)$，由 F_1 和 F_3 可初步判断该包头匹配 R_1。对于整体规则集，在初步判断出包头所匹配的规则后，还需要对匹配结果进行假阳性验证。例如，对于包头 $P'=(1,5,4)$，由 F_1 和 F_3 可初步判断该包头匹配 R_1，而进一步验证后，P' 并不满足 F_2 的取值。

定义 4 规则独立性：对于规则集 χ 中的一条规则 R_i，若对于任意规则 $R_j\in\chi(i\neq j)$，有 $R_i\cap R_j=\varnothing$，则称 R_i 在 χ 中具有独立性。相应地，由相互独立的规则组成的规则集成为独立规则集。

匹配域裁剪要求所操作的规则集具有整体独立性，否则在假阳性验证过程中，会出现多种验证可能，极大增加验证的复杂性。

表 1 的算法 1 给出了对于整体规则集，提取最大独立规则集的算法。算法 1 通过遍历整个规则集，将与规则集中与其他规则没有冲突的规则提取存入独立规则集 G_I 中，最后将剩下的每组相互冲突的子集中选择优先级最低的规则加入 G_I，剩下的规则存入独立集补集 G_D 中。其中，最大独立规则集的求解，本质上是求解最大独立集(Maximal Independent Set, MIS)问题，该问题为图论中的一类 NP 难问题。算法 1 中综合考虑硬件支持方案和 TCAM 中的查表特性，将求解最大独立规则集的算法简化为图的连通子图求解，保证了功能正确性的同时，将算法的时间复杂度减少到了 $O(kN^2)$。

由算法 1 得到的独立规则子集 G_I 后，开始运行匹配域裁剪算法，对匹配域的数量进行裁剪。其中，裁剪过程的域裁剪数量从 1 开始递增，对于每一个域裁剪数量，搜索各种裁剪方案，直至对于某一裁剪数量无可行解。

定义 5 匹配域保留标记向量 $\boldsymbol{S}$：定义 m 维向量 $\boldsymbol{S}$ 为匹配域保留标记向量，其中，$\boldsymbol{S}[i]=1$ 表示匹配域 F_i 保留，$\boldsymbol{S}[i]=0$ 表示匹配域 F_i 被裁减，$|\boldsymbol{S}|$ 表示向量 $\boldsymbol{S}$ 中取值为 1 的分量个数。对于规则集 χ，由 $\boldsymbol{S}$ 确定被裁减的匹配域后，剩下的匹配域组成规则集表示为 $\chi^{\boldsymbol{S}}$，相应地，$\chi^{\boldsymbol{S}}$ 中的规则表示为 $R^{\boldsymbol{S}}$。

表 1 独立规则集搜索算法

```
算法 1 独立规则集搜索算法
输入：χ , N, m
输出：独立规则集 G_I ，独立集补集 G_D
for i = 1 to N
  for j = (i+1) to N
    if R_i ∩ R_j ≠ ∅
      add R_i, R_j to G_D ;
      break;
  add R_i to G_I ;
for every maximal connected subgraph in G_D
      pick the rule R_x with the least priority in χ ;
    add R_x to G_I ;
    remove R_x from G_D ;
return G_I , G_D ;
```

表 2 的算法 2 给出了匹配域裁剪算法执行步骤。算法 2 中主要有 3 层循环，其中最外层循环确定保留的匹配域数量，对匹配域数量从 1 到 m 进行遍历；第 2 层循环在选择特定的匹配域数量 i 前提下，进行匹配域组合的选取，共有 C_m^i 次循环；最内层循环遍历每条规则，确定特定的匹配域组合下能否实现规则的区分。其中，每次有符合条件的匹配域组合后，与当前匹配域组合相比较，保留匹配域最少的方案。该算法的时间复杂度为 $O(m^3N)$。

算法 2 对于独立规则集划分，其首要目标是涵盖所有独立规则集中的规则。对于部分规则集，其不同匹配域取值之间相对的关联性较弱，因此规则

表 2 匹配域裁剪算法

```
算法 2 匹配域裁剪算法
输入： G_I , m
输出：匹配域保留标记 S
Initialize S with value 1;
W_temp = W(χ) ;  //默认初始状态包含所有匹配域
for i = 1 to m
  update_flag = 0;
  for each S_temp whose | S_temp |= i  //遍历所有匹配域组合
    for each two rule R_a^S_temp , R_b^S_temp ∈ χ^S_temp ( a ≠ b )
      if R_a^S_temp ∩ R_b^S_temp ≠ ∅  //若当前组合满足区分度要求，跳出当前循环
        break;
    if W(χ^S) < W_temp  //在符合区分度的组合中选择规则位宽最小组合
      S = S_temp ;  //若当前匹配域组合比原组合更优，则更新组合方案
      updata_flag=1;
  if updata_flag==0
    return S ;
return S ;
```

数量优先的裁剪算法并不能取得较好的裁剪效果。对于小部分规则，将其包含在规则集之外能够明显提高匹配域裁剪的效果。表 3 的算法 3 给出了改进的匹配域裁剪算法，通过设定裁剪后能覆盖的规则数占总数的比例 β 作为阈值，即相对于算法 2 中的最内层循环，算法 3 不要求当前匹配域组合能够区分左右规则，只需能够区分占总数的比例 β 的规则数即可。

表 3 改进的匹配域裁剪算法

算法 3 改进的匹配域裁剪算法

输入：G_I, m, β

输出：匹配域保留标记 $\boldsymbol{S}$

Initialize $\boldsymbol{S}$ with all ones;

$W_{\text{temp}} = W(\chi)$;

for $i = 1$ to m

  update_flag = 0; //若无更新，说明算法结束

  for each $\boldsymbol{S}_{\text{temp}}$ whose $|\boldsymbol{S}_{\text{temp}}| = i$

    collision_count = 0;

    for each two rule $R_a^{\boldsymbol{S}_{\text{temp}}}, R_b^{\boldsymbol{S}_{\text{temp}}} \in \chi^{\boldsymbol{S}_{\text{temp}}}$ ($a \neq b$)

      if $R_a^{\boldsymbol{S}_{\text{temp}}} \cap R_b^{\boldsymbol{S}_{\text{temp}}} \neq \varnothing$

        collision_count = collision_count +1;

      if collision_count> $|G_I| \bullet (1-\beta)$ //冲突规则过多则抛弃当前组合

        Break;

    if $W(\chi^{\boldsymbol{S}}) < W_{\text{temp}}$

      $\boldsymbol{S} = \boldsymbol{S}_{\text{temp}}$;

      updata_flag=1;

  if updata_flag==0

    return $\boldsymbol{S}$;

return $\boldsymbol{S}$;

3　硬件架构

本文提出的基于匹配域裁剪的流表压缩方案主要分为按照逻辑关系进行的匹配域合并和基于统计关系的匹配域裁剪。总体硬件实现方案如图 3 所示。控制器在下发流表之前，流表数据经过中间层进行处理(主要运行第 3 节的相关算法)，将匹配域的合并规则下发到匹配域合并电路相关部分，同时将流表下发到相关 TCAM 部分。其中，匹配域预处理电路按照上层配置的匹配域合并方案设计，负责执行相应匹配域合并操作，从而改变包头域宽度，缩减包头域中实际的匹配域数量。匹配域预处理电路可以由 FPGA、网络处理器等硬件来实现，并且需要结合相应 RAM1 储存信息，用以提取特定匹配域合并时所需信息(例如 IPv4 向 IPv6 地址映射时的前缀码)。包头域经过匹配域合并处理后，送往 TCAM 进行查表匹配操作。其中，TCAM 主要分为两部分，N-TCAM 表项宽度较小，存储经过匹配域裁剪后的流表；W-TCAM 表项宽度为原始流表位宽，主要存储算法 1 中独立集补集 G_D 中的未经裁剪的流表。表项同时送往 N-TCAM 和 W-TCAM 进行查表后，N-TCAM 的匹配结果通过查询 RAM2 中储存的裁剪信息进行假阳性验证。针对可能会出现的两个 TCAM 都有匹配结果的情况，根据算法 1 的独立集提取规则，N-TCAM 和 W-TCAM 中的规则有交集时，W-TCAM 中的表项优先级高于 N-TCAM。优先判决电路根据两部分 TCAM 的优先级提取最优匹配结果进行输出。

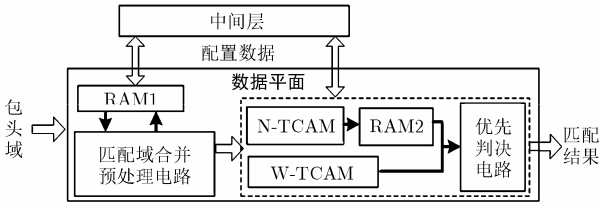

图 3　Field Trimmer 硬件架构

由于常见 TCAM 表项宽度为 36 bit, 72 bit, 144 bit, 288 bit 等，经过匹配域合并和裁剪后，N-TCAM 所需表项宽度相对于原始表项宽度可大幅减小，所选 TCAM 芯片的表项宽度可以减小一半以上。因此，相比于没有裁剪的 TCAM 储存方案，在相同的芯片储存空间下，N-TCAM 可存储更多表项。

4　实验和评估

本文的实验部分主要针对两类规则集进行压缩效果验证。由于 OpenFlow 目前没有公开的大规模应用场景，因此无法获得大规模 OpenFlow 流表进行试验。本文根据国家宽带网络与应用工程技术研究中心的小规模 OpenFlow v1.1 测试流表和根据 ClassBench[22]产生的规则集进行仿真，其中算法运行环境硬件配置为 Intel Core i7-4790 3.6 GHz 处理器，8G 内存；算法运行于 Matlab 2012a, Win7 操作系统。

基于以上实验环境，在规则数 N=5000 的情况下，本文对不同的阈值参数 β 的压缩效果进行了评估。图 4 反映了不同阈值参数 β 下本文方案的压缩效果。由图 4 可以看出，在 β=0.95 的条件下，压缩效果最好。考虑到整体 OpenFlow 规则集统计特征固定，而取样规则满足随机抽取的原则，因此认为规则数为 5000 时的不同阈值参数 β 的压缩效果变化规律能反映出整体环境中 β 对压缩效果影响的变

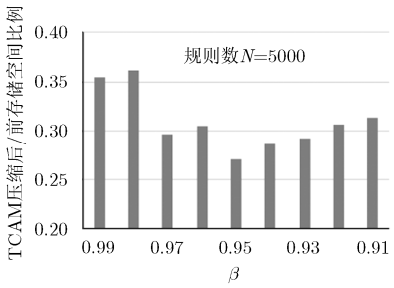

图 4 不同阈值参数 β 选择下的压缩效果对比

化规律，因此得到结论在当前精度下，β 取值 0.95 时能获得最优压缩效果。实际应用中，可根据工程具体需求调整 β 取值精度，达到最优效果。

图 5 给出了压缩后所需储存空间与压缩前所需储存空间的对比。其中，Field Trimmer 方案分别给出了 β=0.99 和 0.95 时的压缩效果。β=0.99 是独立规则子集所占规则总数的近似比例。由图 5 结果显示可以看出，在取最优参数条件下，本文方案所能实现的压缩后储存空间只有原始空间的 20%~30%，是 H-SOFT 实现方案所需 TCAM 存储空间的一半左右。

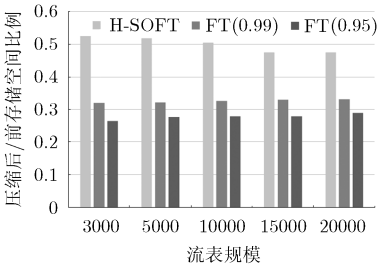

图 5 Field Trimmer(图中简称 FT)与 H-SOFT 对 OpenFlow 流表的储存空间优化性能对比(β 分别取值 0.95 和 0.99)

基于逻辑关系的匹配域合并和基于统计特征的匹配域裁剪都需要按照图 3 硬件架构所示在数据平面之上进行预处理，即算法 1、算法 2 和算法 3 需要一定的运算开销。图 6 显示了本节所述实验环境下 H-SOFT 和 Field Trimmer(FT)的算法运行时间对比。由图可以看出，Field Trimmer 的算法运行时间略高于 H-SOFT，二者的算法运行时间随着流表规模增长的速度近似。

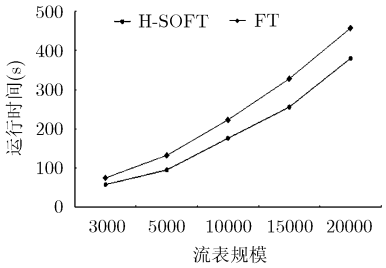

图 6 不同表象规模的算法运行时间对比(以 β =0.99 为例)

为了进一步测试本文方案对于不同规则集的压缩性能，采用了 ClassBench 产生的多种类型的规则集进行了测试。表 4 给出了本文模型对于 ClassBench 产生的 12 中不同规则集的压缩效果。

由表 4 可以看出，对于部分规则集，其匹配域可裁剪性较强，β=0.95 时，acl1 只需一个匹配域即可完成包分类功能；而部分规则集其可裁剪性较弱，β=0.85 时，仍需两个匹配域共 64 bit 长度完成匹配。总体来说，在阈值参数 β 设置 0.9 左右时，Field Trimmer 对于所有规则集都有明显的压缩效

表 4 基于 ClassBench 的不同规则集下 Field Trimmer 压缩效果

规则集名称	规则数量	所需匹配域数量 (所需 bit 位宽)			
		β =1	β =0.95	β =0.9	β =0.85
acl1	9768	4(96)	2(64)	1(32)	1(32)
acl2	9065	5(108)	3(80)	2(64)	1(32)
acl3	9961	4(96)	2(64)	2(64)	1(32)
acl4	9832	4(96)	3(80)	2(64)	2(64)
acl5	9854	4(96)	4(96)	2(64)	2(64)
fw1	9164	5(108)	3(80)	2(64)	2(48)
fw2	9463	4(96)	3(80)	2(64)	2(48)
fw3	9172	5(108)	3(80)	2(64)	2(48)
fw4	9114	5(108)	3(80)	2(64)	2(48)
fw5	9032	5(108)	3(80)	2(64)	2(48)
ipc1	9961	4(96)	3(80)	2(64)	2(48)
ipc2	10000	4(96)	2(64)	2(48)	2(48)

果，此时绝大部分规则储存在 N-TCAM 中，因此，通过适当的参数选择，Field Trimmer 在应用于常见五元组包分类规则集时，也能明显节约 TCAM 的资源消耗。

5　结束语

本文针对以 OpenFlow 为代表的多匹配域包分类规则集提出了一种基于匹配域合并裁剪的优化存储方案。本文方案首先分析多匹配域之间的逻辑关系，通过提取特定的逻辑关系为基础进行匹配域的合并，从而减少表项的匹配域数量；通过对整体规则集特征的分析，对匹配域进行裁剪，裁剪过后以部分匹配域完成匹配功能，从而进一步减少了表项宽度。本文方案在较低的算法复杂度代价下，实现了较好的压缩效果。同时，由于本文压缩方案的实现效果依赖于规则集的特征分析，本文给出了调整参数，方便根据不同的规则集选择最合适的压缩方案。仿真实验结果表明，本文所提出的 Field Trimmer 方案对于 OpenFlow 的流表所需存储空间比 H-SOFT 进一步节省 50%左右；而对于常见的访问控制列表、防火墙等包分类规则集，Field Trimmer 也能够节省 40%以上的 TCAM 存储空间。

参 考 文 献

[1] MCKEOWN N. Software-defined networking[C]. INFOCOM Keynote Talk, Rio de Janero, Brazil, 2009: 30–32.

[2] MCKEOWN N, ANDERSON T, BALAKRISHNAN H, *et al*. OpenFlow: Enabling innovation in campus networks[J]. *ACM SIGCOMM Computer Communication Review*, 2008, 38(2): 69–74. doi: 10.1145/1355734.1355746.

[3] LAKSHMINARAYANAN K, RANGARAJAN A, and VENKATACHARY S. Algorithms for advanced packet classification with ternary CAMs[J]. *ACM SIGCOMM Computer Communication Review*, 2005, 35(4): 193–204. doi: 10.1145/1090191.1080115.

[4] VAMANAN B, VOSKUILEN G, and VIJAYKUMAR T N. EffiCuts: Optimizing packet classification for memory and throughput[J]. *ACM SIGCOMM Computer Communication Review*, 2010, 40(4): 207–218. doi: 10.1145/1851182.1851208.

[5] SONG H and TURNER J S. ABC: Adaptive binary cuttings for multidimensional packet classification[J]. *IEEE/ACM Transactions on Networking*, 2013, 21(1): 98–109. doi: 10.1109/TNET.2012.2190519.

[6] 陈正虎, 兰巨龙, 黄万伟, 等. 一种基于 Bloom-filter 表项压缩的 TCAM 业务识别算法[J]. 电子与信息学报, 2011, 33(9): 2212–2218. doi: 10.3724/SP.J.1146.2011.00058.
CHEN Zhenghu, LAN Julong, HUANG Wanwei, *et al*. A TCAM service identification algorithm based on access compression using bloom-filter[J]. *Journal of Electronics & Information Technology*, 2011, 33(9): 2212–2218. doi: 10.3724/SP.J.1146.2011.00058.

[7] BREMLER-BARR A and HENDLER D. Space-efficient TCAM-based classification using Gray coding[J]. *IEEE Transactions on Computers*, 2012, 61(1): 18–30. doi: 10.1109/TC.2010.267.

[8] BREMLER-BARR A, HAY D, and HENDLER D. Layered interval codes for TCAM-based classification[C]. INFOCOM 2009, Rio de Janero, Brazil, 2009: 1305–1313. doi: 10.1109/INFCOM.2009.5062045.

[9] MEINERS C R, LIU A X, and TORNG E. Bit weaving: A non-prefix approach to compressing packet classifiers in TCAMs[J]. *IEEE/ACM Transactions on Networking*, 2012, 20(2): 93–102. doi: 10.1109/TNET.2011.2165323.

[10] WEI Rihua, XU Yang, and CHAO H J. Finding nonequivalent classifiers in Boolean space to reduce TCAM usage[J]. *IEEE/ACM Transactions on Networking*, 2016, 24(2): 968–981. doi: 10.1109/TNET.2015.2402093.

[11] MISHRA T, SAHNI S, and SEETHARAMAN G. PC-DUOS: Fast TCAM lookup and update for packet classifiers[C]. IEEE Symposium on Computers and Communications, Kerkyra, Corfu, Greece, 2011: 265–270. doi: 10.1109/ISCC.2011.5983851.

[12] BANERJEE T, SAHNI S, and SEETHARAMAN G. PC-DUOS+: A TCAM architecture for packet classifiers[J]. *IEEE Transactions on Computers*, 2014, 63(6): 1527–1540. doi: 10.1109/TC.2012.287.

[13] BANERJEE T, SAHNI S, and SEETHARAMAN G. PC-TRIO: A power efficient TCAM architecture for packet classifiers[J]. *IEEE Transactions on Computers*, 2015, 64(4): 1104–1118. doi: 10.1109/TC.2014.2315645.

[14] CHANG D Y and WANG P C. TCAM-based multi-match packet classification using multidimensional rule layering[J]. *IEEE/ACM Transactions on Networking*, 2016, 24(2): 1125–1138. doi: 10.1109/TNET.2015.2411274.

[15] SCHRIFF L, AFEK Y, and BREMLER-BARR A. Orange: Multi field OpenFlow based range classifier[C]. ACM/IEEE Symposium on Architectures for Networking & Communications Systems, Oakland, CA, USA, 2015: 63–73. doi: 10.1109/ANCS.2015.7110121.

[16] KOGAN K, NIKOLENKO S I, ROTTENSTREICH O, *et al*. Exploiting order independence for scalable and expressive packet classification[J]. *IEEE/ACM Transactions on Networking*, 2016, 24(2): 1251–1264. doi: 10.1109/TNET.

2015.2407831.

[17] LIU Z, WANG X, YANG B, *et al*. BitCuts: Towards fast packet classification for order-independent rules[J]. *ACM SIGCOMM Computer Communication Review*, 2015, 45(4): 339–340. doi: 10.1145/2785956. 2789998.

[18] GE J, CHEN Z, WU Y, *et al*. H-SOFT: A heuristic storage space optimisation algorithm for flow table of OpenFlow[J]. *Concurrency & Computation Practice & Experience*, 2014, 27(13): 3497–3509. doi: 10.1002/cpe.3206.

[19] YUN R Q and VIKTOR K P. High-performance and dynamically updatable packet classification engine on FPGA[J]. *IEEE Transactions on Parallel and Distributed Systems*, 2016, 27(1): 197–209. doi: 10.1109/TPDS.2015. 2389239.

[20] RFC2460[OL]. http://www.rfc-editor.org/info/rfc2460. 2016.

[21] RFC6052[OL]. http://www.rfc-editor.org/info/rfc6052. 2016.

[22] TAYLOR D E and TURNER J S. ClassBench: A packet classification benchmark[J]. *IEEE/ACM Transactions on Networking*, 2005, 3(3): 499–511. doi: 10.1109/TNET.2007. 893156.

孙鹏浩： 男，1992 年生，博士生，研究方向为新型网络体系结构.

兰巨龙： 男，1962 年生，教授，博士生导师，主要研究方向为网络体系结构、信息安全.

陆肖元： 男，1975 年生，高级工程师，主要研究方向为网络体系结构、光通信技术.

胡宇翔： 男，1982 年生，博士，主要研究方向为新型网络体系结构、网络安全.

马 腾： 男，1989 年生，博士生，研究方向为新型网络体系结构.