DOI：10.7652/xjtuxb201802002

面向软件定义网络的流表优化方案

唐亚哲[1]，张永琪[1]，颜自坚[2]，朱桂英[2]
（1.西安交通大学电子与信息工程学院，710049，西安；2.中国电力科学院南京分院，210000，南京）

摘要：针对目前软件定义网络中细粒度的流匹配机制造成的网络流表项空间开销和查询开销爆炸式增长等问题，提出了一种全新的基于布隆过滤器(Bloom Filter)的多级流表结构。该结构为混合流表结构，采用Bloom Filter多级流表结构来存储流表项，主要着眼于提高软件定义网络(SDN)交换机流表的容量和加快流表项的匹配速度；在流表项语义层面，设计并实现了控制器与SDN交换机之间的中间适配层模块来解决语义冲突问题。基于真实流量的实验结果表明，在规则占用空间上，与传统流表相比，Bloom Filter在流表越精细的情况下优化比率越高，最高可达90.7%。随着流表项规则的增加，匹配耗时优化效率提高，匹配时间最多可减少99.4%。该问题的解决可望为SDN网络的大规模实用化部署奠定数据层面的基础。

关键词：软件定义网络；混合流表；布隆过滤器；多流表优化
中图分类号：TP393.02 **文献标志码**：A **文章编号**：0253-987X(2018)02-0013-05

A Flow Table Optimization Scheme for Software Defined Network

TANG Yazhe[1], ZHANG Yongqi[1], YAN Zijian[2], ZHU Guiying[2]
(1. School of Electronic and Information Engineering, Xi'an Jiaotong University, Xi'an 710049, China;
2. Nanjing Branch of China Electric Power Research Institute, Nanjing 210000, China)

Abstract: A new multilevel flow table architecture based on Bloom Filter is proposed to solve the problem of the explosive growth of the flow table's space occupation and look-up overhead due to the fine grained flow matching mechanism in SDN networks. This architecture follows the hybrid flow table paradigm and uses a Bloom Filter pipeline to store flow entries. The motivation is to construct secondary flow tables with very high capacity and to speed up the matching speed of flow table items. An intermediate adaptation layer between controllers and OpenFlow switches is designed to transform the corresponding messages and to avoid possible semantic conflicts. Experiments results show that when the flow table becomes more fine-grained, the space occupation of the proposed scheme is more effectively reduced, up to 90.7%, and when the number of the flow entries increases, the packet matching time is greatly reduced and the maximum reduction of matching time is 99.4%. The solution of the problem is expected to lay a solid foundation in data plane for the large-scale practical deployment of SDN networks.

Keywords: software defined network; hybrid flow table; Bloom Filter; multi-flow table optimization

收稿日期：2017-06-14。 作者简介：唐亚哲(1970—)，男，副教授，博士生导师。 基金项目：国家自然科学基金资助项目(61672425)；陕西省自然科学基础研究计划资助项目(2016JM6066)；国家电网公司科学技术资助项目(DZ71-16-030)。
网络出版时间：2017-11-28 网络出版地址：http://kns.cnki.net/kcms/detail/61.1069.T.20171128.1540.008.html

软件定义网络(SDN)[1-3]是一种新型网络体系结构,它将控制平面与数据平面解耦和,具有精细的流管理属性。精细的流管理虽然可以更加精准地定义网络,但是也带来了控制规则及其内存需求的爆炸性增长。目前,大部分的 SDN 硬件交换机将控制规则部署在三态内容寻址寄存器(TCAM)中,但 TCAM 非常昂贵且耗电量大,因此流表的可扩展性问题,已经成为制约 SDN 网络广泛部署和应用的主要因素之一。

为了更好地解决流表项的可扩展性问题,目前的研究大致可以分为以下几类方案:①对流表本身进行优化,设计软硬件结合的流表结构进行扩容,文献[4-6]利用 TCAM 和其他存储容器设计出了新型混合流表结构,这种结构将经常使用的流表项存储在 TCAM 中,其他的细粒度流表项存储在其他内存中;②通过压缩流表项的大小和数量,使原始流表可以容纳更多的流表项,文献[7-10]对网络拓扑进行分层,边缘层使用精确匹配,中心层进行标签聚合匹配;③文献[11]根据负载因子动态调整交换机流表中流表项的停滞超时时间,来保证流表项的空间与时间上的动态平衡;④文献[12]利用 SDN 控制器网络视图将端到端路径信息编码到包地址中。

本文基于布隆过滤器(Bloom Filter)[13]设计了新型的细粒度流表项存储结构,在可接受的匹配速度范围内降低细粒度流表项的内存需求和几乎恒定的流表项查询时间。为了使新的流表结构适应于不同的控制器并保持语义一致性,设计并实现了控制器和 SDN 交换机之间的中间适配层,以转换相应的 OpenFlow 消息。

1 系统架构

本文提出的新型流表系统结构如图 1 所示。在数据平面设计新型混合多流表结构,结合 TCAM 与 Bloom Filter 共同存储流表规则。Bloom Filter 是一种空间效率很高的随机数据结构,利用位数组可以很简洁地表示一个集合,并能快速判断一个元素是否属于这个集合。TCAM 成本昂贵,查找速度快;Bloom Filter 成本低,空间效率高,二者配合能够实现优势互补。通过对规则占用空间的计算,将范围匹配较大的规则或者高频流规则放置在 TCAM 中进行匹配,将精细匹配的规则放置在 Bloom Filter 中匹配,达到充分利用空间、节省硬件成本的目的。

在新型流表系统中,当新的数据包到达交换设备,如果没有规则匹配,则触发数据报文上报行为将数据包发送到控制器,控制器生成规则之后通过 FlowMod 消息下发。中间层通过与控制器建立的连接先收到 FlowMod 消息,通过解析、计算等一系列转化生成新的流表添加消息,通过与交换机的连接发送至交换机解析并添加规则,这样数据包便能够通过规则查找进行匹配并执行相应的动作。

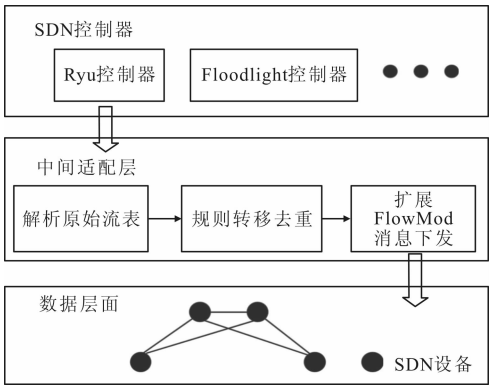

图 1 新型流表系统结构示意图

2 混合多流表结构设计

新型流表结构是多级流表结构,该结构的第一级是 TCAM 流表,连接多级 Bloom Filter 流表。每级 Bloom Filter 多流表根据处理动作划分,原始流表中具有相同动作的规则放在同一级 Bloom Filter 流表中。每一级 Bloom Filter 流表分为匹配域存储及动作执行 2 个部分。匹配域存储部分由多个 Bloom Filter 位图组成,每个位图负责各自匹配域组合,数据包进入流表时,只取相关的包头部分进行哈希值计算,一旦匹配成功则进入动作执行部分执行动作。多级 Bloom Filter 结构如图 2 所示,每级 Bloom Filter 流表分为 $B_1,\cdots,B_n$ 多个 Bloom Filter 子表;$P_1,\cdots,P_n$ 为不同的匹配域组合。

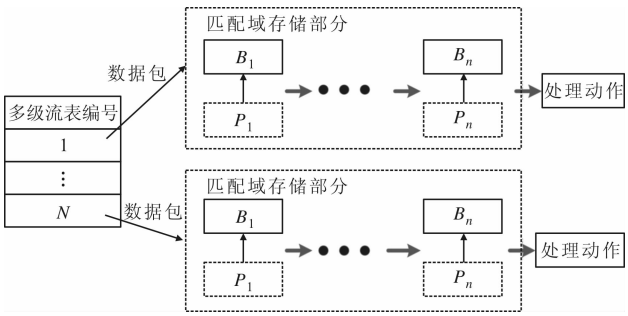

图 2 多级 Bloom Filter 结构示意图

数据包进入交换机后先在 TCAM 中进行规则查找,如果规则匹配成功,就执行相应的动作;如果一直匹配不成功,则默认将数据包传到第一级

Bloom Filter 中进行逐个匹配。当数据包经过某个 Bloom Filter 时，该 Bloom Filter 只在数据包包头提取自己匹配域存储部分设定的匹配域的值，匹配成功即执行动作；匹配失败则进入下一个 Bloom Filter 流表中进行重复的匹配流程。

3　流表中间适配层

OpenFlow 规则主要由匹配域和优先级表征，先查看优先级高的规则，如果优先级高的规则失配才查看优先级低的规则，以匹配域与优先级的共同作用来表征规则的语义。本文根据规则间的几种不同关系进行处理，进行等价的语义转义使得规则在 Bloom Filter 多流表中与在原始流表中等价。

流表中间适配层的规则转义与整合过程如下。

步骤 1　将每一个流表项放入不同的子表中，按照优先级从高到低遍历原始流表的每一条流表项，流表项的各个匹配域非通配，则表示该匹配域有值。流表项按照匹配域的组合情况进行分类，如图 3 所示。该阶段将不考虑通配符的匹配字段。

步骤 2　将规则按照语义进行等价转义。每个表的匹配域组合情况存放在一个集合中，遍历这个集合，两两比较规则匹配域组合及值，将规则按规则间语义无关、规则语义部分重叠、规则匹配域完全覆盖 3 种关系进行处理。

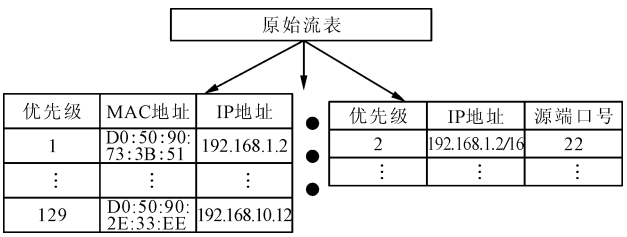

图 3　原始流表按照匹配域组合分类成多个子表

本文采用二元组集合来表征每条流表项的匹配域。例如在表 1 中，本文定义交换机入端口号为 P_{in}，源 IP 地址为 S_{ip}，源端口号为 S_{p}。F_1 和 F_2 的匹配域集合分别为 $A=\{(P_{\text{in}},1),(S_{\text{ip}},192.168.1.2)\}$ 和 $B=\{(S_{\text{ip}},192.168.1.2),(S_{\text{ip}},22)\}$。

规则语义无关是指规则间匹配域内容不重叠，遍历顺序即使与优先级约定的不一致也不会导致对数据包的错判，数据包绝对不会同时匹配到这 2 条流表项，这种情况下不需要进行转换。

表 1　规则语义部分重叠示例

流表项名称	优先级	交换机入端口号	源 IP 地址	源端口号	执行动作
F_1	2	1	192.168.1.2	(0, 65535)	转发
F_2	1	(0, 65535)	192.168.1.2	22	丢弃

规则语义部分重叠是指 2 个表之间的匹配域组合情况存在部分交叠，并且交叠部分的匹配域的值相同。存在一个数据包同时匹配到这 2 条流表项，发生冲突。

从集合的角度来看，规则语义部分重叠的转义分 3 个步骤：①获取 2 个流表项的匹配域的公共部分；②通过集合差运算获得具有较高优先级的流表项中匹配域的唯一部分；③从第②步中获得的唯一匹配域中选择一个匹配域（例如 (f,v)），并将其添加到 $(f,!v)$ 形式的较低优先级的流表项的匹配字段中。

规则匹配域完全覆盖的解决办法类似于部分覆盖，分为 2 个步骤：①通过集合差运算获得具有较高优先级的流表项的匹配域的唯一部分，即 $B-A$，由于 $B\supset A$，所以 $B-A=B-B\cap A$；②在第①步中获得的唯一匹配域中选择一个匹配域（例如 (f,v)），并将其添加到 $(f,!v)$ 形式的较低优先级流表项的匹配域中。

步骤 3　对规则匹配域覆盖的情况，进行去重工作。优先级加匹配域的共同作用，有些规则可能永远无法查看到。这样的规则下发到流表中没有意义，可以在中间层去重以达到进一步优化流表空间的目的。

4　实验与测试

本文搭建了多级流表测试系统，选择 Ryu 控制器和 Floodlight 控制器作为 SDN 控制器。通过修改和替换部分 CPqD 源代码，搭建出基于 CPqD OpenFlow1.3 软件的 OpenFlow 交换机。主要工作包括：①修改 lib 模块，支持 OpenFlow 实验信息的交换；②用新的 Bloom Filter 多级流表替换原始流表；③更换超时检测逻辑和计数器逻辑。在此基础上进行功能和性能评估的实验。由于篇幅的限制，本文主要进行了性能测试。由于 Bloom Filter 交换机基于 CPqD 软件交换机开发，因此所有的性能测试都是针对 CPqD 软件交换机。

4.1　占用空间实验

流表配置接口进行流表规则的配置，选取 CPqD 开源 OpenFlow1.3 交换机为对比对象，在 Bloom Filter 多流表交换机、CPqD 交换机中进行规则添加。本实验选用 IP 五元组（源 IP 地址、源端

口、目的 IP 地址、目的端口、传输层协议)进行流表匹配域的组合。

对于任一条规则,随着匹配域组合位宽的增加,该规则占用空间的变化情况如图 4 所示。在 CPqD 交换机中,一条规则占用的位宽随着规则匹配域位宽的增加呈线性增加,而 Bloom Filter 的一条规则占用的位图宽度并不会因为匹配域的增多或者减少而变化,更具有可扩展性。

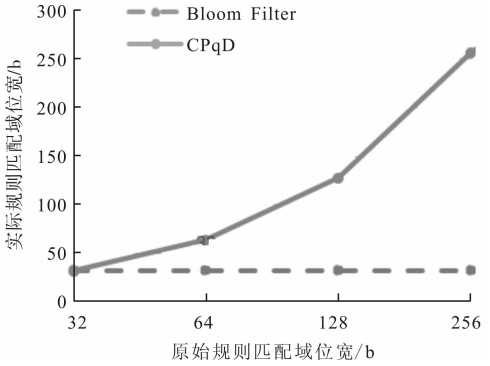

图 4 CPqD 交换机与 Bloom Filter 多级流表的一条规则占用空间对比

经过统计计算得到,在各个位宽情况下,对于一条规则占用的空间,Bloom Filter 流表的空间优化比率为

$$\eta = (M_C - M_B)/M_C \tag{1}$$

式中:M_C 表示 CPqD 交换机单条规则占用的空间;M_B 表示基于 Bloom Filter 的多级流表结构中单条规则占用的空间。

SDN 经常进行精细的基于流的匹配,运用于如细粒度流量识别、带宽管理等场景。表 2 中 Bloom Filter 流表在规则匹配域位宽增加的情况下优化比率提高,最高可达 90.7%,在其他匹配域情况下优化比率也在 48%以上(在 32 b 情况下没有优化空间是因为 Bloom Filter 的一条规则不论匹配域宽度为多少,占用的空间都是恒定在 33 b)。

表 2 Bloom Filter 流表一条规则占用空间优化比率

规则匹配域位宽/b	优化比率/%
32	0.0
64	48.4
128	74.2
256	87.1
512	90.7

4.2 数据包匹配处理耗时测试

测试时,数据包发生器分别发送流量到 Bloom Filter 多级流表、CPqD 多级流表中,并针对 Bloom Filter 多级流表、CPqD 多级流表添加相同的规则和规则匹配域,规则的执行动作部分都为丢弃操作。由于仅进行查找时间统计,因此数据包匹配成功则进行丢弃处理。2 种交换机中都进行如下时间统计

$$T_N = T_D - T_E \tag{2}$$

式中:T_N 表示数据包匹配处理总时间;T_D 表示数据包匹配完毕并执行动作的总时间;T_E 表示数据包进入交换机的耗时。在 CPqD 源码及 Bloom Filter 多级流表都加入相应的时间戳进行计算,统计后求得平均值。

图 5 为 CPqD 交换机流表匹配处理耗时的统计结果,可见随着流表数的增大,CPqD 交换机查找流表的时间大大增加。随着规模的增大,Bloom Filter 多级流表在同等规模的规则数条件下仍保持 3~5 μs 的查找时间。

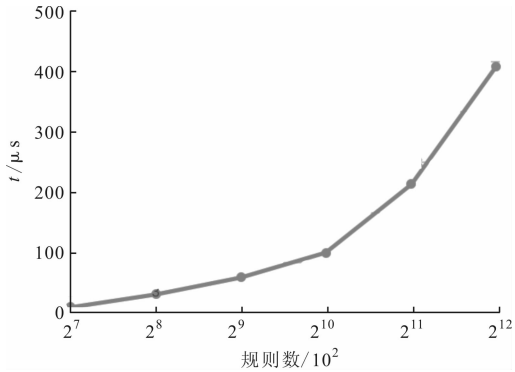

图 5 CPqD 交换机流表匹配处理耗时统计结果

在不同规则规模下,Bloom Filter 多级流表的匹配耗时优化比率为

$$\eta = (T_C - T_B)/T_C \tag{3}$$

式中:T_C 表示 CPqD 交换机的匹配耗时;T_B 表示基于 Bloom Filter 的多级流表结构的匹配耗时。

表 3 给出了 Bloom Filter 流表数据包匹配耗时优化比率,由表 3 可以看出,Bloom Filter 多级流表的数据包匹配耗时优化效果明显,最多可达 99.4%,极大地优化了数据包处理的耗时,提高了匹配流程的

表 3 Bloom Filter 流表数据包匹配耗时优化比率

规则数/10^2	优化比率/%
1	88.0
5	96.0
10	98.1
20	98.7
50	99.2
100	99.4

吞吐率。

本次实验针对的是 SDN 软件交换机的相关性能指标，与采用多级 TCAM 架构的 SDN 硬件交换机相比，Bloom Filter 查询效率仍存在一定的差距，但是从软件交换机（在云体系中大量使用）的优化角度而言，多级 TCAM 昂贵、耗电的缺点使其无法大规模部署，而基于 Bloom Filter 的多级流表很好地在实际部署与性能之间实现了平衡，便于大规模应用。

5　结　语

SDN 流表优化系统集合了 Bloom Filter 多级流表与 TCAM，在中间适配层进行语义转义并解决了语义冲突问题。基于 CPqD 软件交换机进行设计和实验，实验结果表明空间优化率最多可达 90％以上，数据包匹配处理耗时优化率最多可达 88％～99％以上，达到了双重优化的目的。

针对 Bloom Filter 的假阳性问题，本文将误判率限定在 10^{-7}，用以降低误判带来的影响。同时，在后续的工作中本系统将引入冲突检测和消除机制，采用多级流表顺序匹配的方法，该方法通过判定匹配次数来检测是否发生误判。后续工作增加冲突链表备份机制来消除误判，虽然这在一定程度上增加了查询开销和存储开销，但考虑到极低的假阳性概率，这些开销可以忽略不计。

参考文献：

[1] 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.

[2] 左青云，陈鸣，赵广松，等. 基于 OpenFlow 的 SDN 技术研究 [J]. 软件学报，2013，24(5)：1078-1097.
ZUO Qingyun, CHEN Ming, ZHAO Guangsong. Research on OpenFlow-based SDN technologies [J]. Journal of Software, 2013, 24(5): 1078-1097.

[3] VAUGHAN-NICHOLS S J. OpenFlow: the next generation of the network? [J]. Computer, 2011, 44(8): 13-15.

[4] CURTIS A R, MOGUL J C, TOURRILHES J, et al. DevoFlow: scaling flow management for high-performance networks [C]// Proceedings of the 2011 ACM SIGCOMM Conference. New York, USA: ACM, 2011: 254-265.

[5] KATTA N, ALIPOURFARD O, REXFORD J, et al. CacheFlow: dependency-aware rule-caching for software-defined networks [C]// Proceedings of the Symposium on SDN Research. New York, USA: ACM, 2016: 6.

[6] LI X, XIE W. CRAFT: a cache reduction architecture for flow tables in software-defined networks [C]// Proceedings of the 2017 IEEE Symposium on Computers and Communications. Piscataway, NJ, USA: IEEE, 2017: 967-972.

[7] KANAK A, COLIN D, ERIC R. Shadow MACs: scalable label-switching for commodity ether-net [C]// Proceedings of the Workshop on Hot Topics in Software Defined Networking. New York, USA: ACM, 2014: 157-162.

[8] ARNE S, HOLGER K. Using MAC addresses as efficient routing labels in data centers [C]// Proceedings of the Workshop on Hot Topics in Software Defined Networking. New York, USA: ACM, 2014: 115-120.

[9] CASADO M, KOPONEN T, SHENKER S, et al. Fabric: a retrospective on evolving SDN [C]// Proceedings of the First Workshop on Hot Topics in Software Defined Networks. New York, USA: ACM, 2012: 85-90.

[10] IYER A S, MANN V, SAMINENI N R. SwitchReduce: reducing switch state and controller involvement in OpenFlow networks [C]// Proceedings of the IEEE IFIP Networking Conference. Piscataway, NJ, USA: IEEE, 2013: 1-9.

[11] 史少平，庄雷，杨思锦. 一种基于预测与动态调整负载因子的 SDN 流表优化算法 [J]. 计算机科学，2017，44(1)：123-127.
SHI Shaoping, ZHUANG Lei, YANG Sijin. SDN optimization algorithm based on prediction and dynamic load factor [J]. Computer Science, 2017, 44(1): 123-127.

[12] ALGHADHBAN A, SHIHADA B. Energy efficient SDN commodity switch based practical flow forwarding method [C]// Proceedings of the Network Operations and Management Symposium. Piscataway, NJ, USA: IEEE, 2016: 784-788.

[13] ANDREI B, MICHAEL M. Network applications of bloom filters: a survey [J]. Internet Mathematics, 2004, 1(4): 485-509.

（编辑　武红江）