Replication Free Rule Grouping for Packet Classification

Xiang Wang[1, 2], Chang Chen[1, 2] and Jun Li[2, 3]
[1]Department of Automation, Tsinghua University, China
[2]Research Institute of Information Technology, Tsinghua University, China
[3]Tsinghua National Lab for Information Science and Technology, China
{xiang-wang11, chenc09} @ mails.tsinghua.edu.cn, junl@tsinghua.edu.cn

ABSTRACT

Most recent works demonstrate that grouping methodology could bring significant reduction of memory usage to decision-tree packet classification algorithms, with insignificant impact on throughput. However, these grouping techniques can hardly eliminate rule-replication *completely*. This work proposes a novel rule grouping algorithm *without any* replication. At each space decomposition step, all rules projecting on the split dimension form the maximum number of non-overlapped ranges, which guarantees the modest memory usage and grouping speed. Evaluation shows that the proposed algorithm achieves comparable memory size with less pre-processing time.

Categories and Subject Descriptors

C.2.3 [**Computer-Communication Networks**]: Network Operations – *Network management, Network monitoring*

General Terms

Algorithms, Design, Performance

Keywords

Packet Classification, Algorithms, Rule Replication

1. INTRODUCTION

Packet classification has been well studied over the past couple of decades. Most decision-tree algorithms trade memory usage for throughput, and employs replication to avoid backtracking search. However, replication will cause critical problems in some specific application scenarios:

- *Excessive overhead in memory size of classifiers that contain a number of wildcard rules*: cutting the wildcard rules usually induce considerable replication.
- *Long pre-processing time*: all replicated rules need to be classified in their subspaces respectively, leading to more internal node processing.
- *Almost impossible to incrementally update the search data structure*: the system complexity rises due to indices to all replicated rules in many sub-trees.

SIGCOMM'13, August 12–16, 2013, Hong Kong, China.

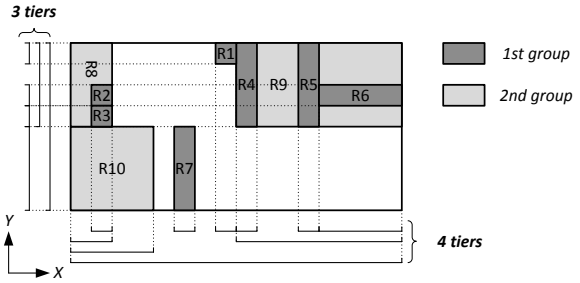

Figure 1. Example 2D-classifier and grouping result

These shortages limit the practical usage of decision-tree algorithms in resource restricted commercial products and dynamically changing network topologies, e.g. virtual switches in software-defined networks (SDN) [1]. Two of the most advanced algorithms have shown the superiority of rule set partition in the improvement of memory performance. EffiCuts [2] defines the separability to categorize rules into small and large range on each dimension. This approach faces difficulties in replication elimination in multi-tiered overlapped classifiers. Figure 1 shows a 2-dimension classifier with 10 rules. It is obvious that rules with multiple different ranges are difficult to be separated into two categories without any overlap. Besides, the separation of small and large range rules may reduce classification efficiency if the large-range rules are not overlapped. ParaSplit [3] combines clustering and intelligent optimization algorithms to reduce memory size, but takes numerous iterations to converge.

This work investigates the overlap of classifiers, and proposes an efficient rule grouping algorithm that guarantees replication free, with both memory size and pre-processing time improved.

2. ALGORITHM

Since the cutting of overlapped rules will potentially cause replication, the basic idea of the replication free algorithm is to group rules where the decision-tree building procedure does not cut *any* rules[1] at *all* space decomposition steps.

The simple and conservative method is to prohibit rule splitting during the iterations of the entire decision-tree building procedure. However, this method will take much time to converge. Because many decision-tree algorithms employ heuristics of current rule set to determine the cut dimension and cut point(s) at each step, the exclusion of rule splitting will impact the calculation of heu-

[1] except for the default rule that covers the entire search space

ristic information and force the building procedure to restart from the tree root.

To achieve fast pre-processing, a greedy and aggressive approach is adopted to avoid replication proactively. The entire rule set is initially selected to execute the following iteration, and the rules excluded in the current iteration form the new group rule set to be processed next. The iteration takes the following steps:

STEP-1: Building the overlap hierarchy
The overlap hierarchy of classifiers is built for each dimension. All rules in the same tier of the specific dimension are non-overlapped with each other, and the hierarchy is built from the small-range rules at the top to the large-range rules at the bottom.

STEP-2: Selecting the dimension of maximum distinction
After building the overlap hierarchy, the topmost tier has the undivided ranges on each dimension. In order to get fast classification speed, the dimension of maximum number of undivided ranges with more rules is selected. During the first iteration on the example classifier, **X** dimension where 6 ranges with 7 rules (**R1** ~ **R7**) are formed is firstly select.

STEP-3: Excluding the overlapped rules
The rules below the topmost tier of the selected dimension are greedily removed from the current rule set. The exclusion does not induce the restart of iteration, since the cut decision conducted in the second step only involves the rules in the topmost tier.

STEP-4: Classifying rules of the same undivided range on other dimensions
The rules that project to the same undivided range on the dimension selected above need to be classified on the other dimensions. The procedure carries out the upper three steps, until all rules are classified or all dimensions are examined. After cutting on **X** dimension of the example classifier, **R2** and **R3** which have the same range on **X** dimension can be classified on **Y** dimension completely. When the current iteration stops, the remaining rules compose one group.

3. EVALUATION

Evaluations of memory size and pre-processing time are conducted on an Intel i3-2310M platform with classifiers generated by classbench[1]. The rule set type covers Access Control List (ACL), Firewall (FW) and IP Chain (IPC). The scale of these classifiers ranges from 1K to 10K. The grouping algorithm is implemented in Python to fast verify our idea (implying the improvement of pre-processing time when using C implementation). HyperSplit [4] is employed to build the decision-tree.

Table 1 compares the memory sizes of HyperSplit without grouping, EffiCuts with only "separable trees" grouping, ParaSplit with 100K iterations and the proposed Replication Free Grouping (RF_GRP). Compared with HyperSplit, all grouping methods have less memory size, with certain exceptions on simple ACL classifiers for both EffiCuts and ParaSplit. Although ParaSplit achieves less memory size than EffiCuts, it exhibits the long convergence time and the possibility of falling into the local optimum on most classifiers where RF_GRP outperforms. Table 2 shows the pre-processing time of RF_GRP, consisting of the grouping time (RF_GRP_Group) and the data structure building time (RF_GRP_Build). It is observed that the overall pre-processing time is improved for most complex classifiers.

[1] http://www.arl.wustl.edu/classbench/

Table 1. Memory size (KB) comparison

Rule Set	HyperSplit	EffiCuts	ParaSplit	RF_GRP
ACL_1K	85	99	81	85
ACL_5K	437	470	428	382
ACL_10K	947	1053	995	928
FW_1K	3569	134	50	56
FW_5K	249134	937	412	404
FW_10K	1008118	2669	857	864
IPC_1K	1492	99	82	78
IPC_5K	25637	492	392	341
IPC_10K	66837	986	823	707

Table 2. Pre-processing (ms) for HyperSplit and RF

Rule Set	HyperSplit	RF_GRP_Group	RF_GRP_Build
ACL_1K	72	*1158*	29
ACL_5K	586	*2758*	150
ACL_10K	2370	*12633*	773
FW_1K	1324	*1239*	21
FW_5K	93544	*5920*	287
FW_10K	271159	*26286*	1323
IPC_1K	503	*1268*	24
IPC_5K	13302	*3008*	153
IPC_10K	35874	*7210*	496

4. CONCLUSION AND FUTURE WORK

Facing with the new challenges of constrained resource and the demands for fast update in emerging SDN applications, this work proposed a packet classification rule grouping algorithm optimized for both memory size and pre-processing time. It guarantees replication free in decision-tree, and supports reconstruction of local data structure for fast update. To alleviate the deterioration of throughput caused by the increment of memory access, a multi-way tree variation of HyperSplit with CPU instruction acceleration has been designed and under development.

5. REFERENCES

[1] J. Pettit, J. Gross, B. Pfaff, M. Casado and S. Crosby. "Virtual Switching in an Era of Advanced Edges", in Proc. of DC CAVES, 2010.

[2] B. Vamanan, G. Voskuilen and T. Vijaykumar. "EffiCuts: Optimizing Packet Classification for Memory and Throughput", in Proc. of SIGCOMM, 2010.

[3] J. Fong, X. Wang, Y. Qi, J. Li and W. Jiang. "ParaSplit: A Scalable Architecture on FPGA for Terabit Packet Classification," in Proc. of HOTI, 2012.

[4] Y. Qi, L. Xu, B. Yang, Y. Xue and J. Li. "Packet Classification Algorithms: From Theory to Practice", in Proc. of INFOCOM, 2009.