Spectral Clustering Based Regular Expression Grouping

Zhe Fu[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
fu-z13@mails.tsinghua.edu.cn, junl@tsinghua.edu.cn

ABSTRACT

Regular expression matching has been playing an import role in today's network security systems with deep inspection function. However, compiling a set of regular expressions into one Deterministic Finite Automata (DFA) often leads to state explosion, which means huge or even impractical memory cost. Distributing regular expressions into several groups and building DFAs independently has been proved an efficient solution, but the previous grouping algorithms are either locally optimal or time-consuming. In this work, we proposed a new grouping method based on Spectral Clustering, which defines the *similarity* between regular expressions and then transforms grouping problem to clustering problem. Preliminary experiments illustrate that our grouping algorithm achieves efficient result with much less processing time.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General – *Security and protection (e.g., firewalls)*

General Terms

Algorithms, Performance, Design, Experimentation.

Keywords

Regular expression matching; deep inspection; DFA; grouping.

1. INTRODUCTION

Nowadays, deep inspection is widely used to identify and filter network traffic. The payload of network packet is examined for virus, spam, intrusions or other criteria which is described as defined signatures. With high flexibility and expressiveness, regular expression matching is playing an important part in network deep inspection systems.

To perform regular expression matching, a set of regular expressions is first compiled to nondeterministic finite automata (NFA), which has less states. However, the space-efficiency of NFA is at the cost of a mass of state transitions to process for each input character. As a result, it is generally necessary to further convert NFA into DFA, which is deterministic finite automata. DFA only requires precisely one state transition lookup per input character (O(1) time complexity), so DFA is always fast and becomes the prior choice in practical deep inspection systems.

Unfortunately, DFA needs more states and transitions compared to NFA, and in many cases suffers from the state explosion. To achieve fast regular expression matching in practice, varieties of techniques such as DFA compression and hardware acceleration have been proposed. Regular expression grouping [1-4] is a newly proposed method and it is very applicable to parallel computation. In this work, we present a fast and efficient grouping algorithm to address the problem of state explosion of DFA.

ANCS'14, October 20–21, 2014, Los Angeles, CA, USA.
ACM 978-1-4503-2839-5/14/10.
http://dx.doi.org/10.1145/2658260.2661771

2. ALGORITHM

Aiming at distributing a set of regular expressions into several subsets, regular expression grouping can be regarded as a clustering problem. Many sound clustering method have been put forward in machine learning and pattern recognition field. However, one regular expression cannot be considered as a sample point since it has no conventional *similarity* to others.

The basic idea of clustering-based grouping algorithm is to define the *similarity* between regular expressions. If the states number of DFA compiled from regular expression r_i is S_i, and the states number of DFA compiled from regular expression pair r_i and r_j is $S_{i,j}$, then the *similarity* between regular expression r_i and r_j can be defined as follows:

$$Similarity(i,j) = \frac{S_i + S_j}{S_{i,j}} \tag{1}$$

Larger similarity means that regular expression r_i and r_j are better to be distributed into the same group, where they are less likely to cause the state explosion. To achieve fast and efficient grouping result, we take advantage of the core idea of Spectral Clustering, which is a clustering algorithm based on graph theory and makes use of the similarity relation among sample points to cluster.

Specifically, for a grouping problem of distributing *N* regular expressions into *k* groups, the Spectral Clustering based regular expression grouping algorithm has following four steps.

STEP-1: Calculating the similarity matrix

Using definition (1), we can calculate the *similarity* between each two regular expressions of a rule set. In particular, for a rule set with *N* regular expressions, an *N* by *N* adjacent matrix W is generated.

STEP-2: Calculating the unnormalized graph Laplacian matrix L

L is calculated by D – W, where D is a degree matrix.

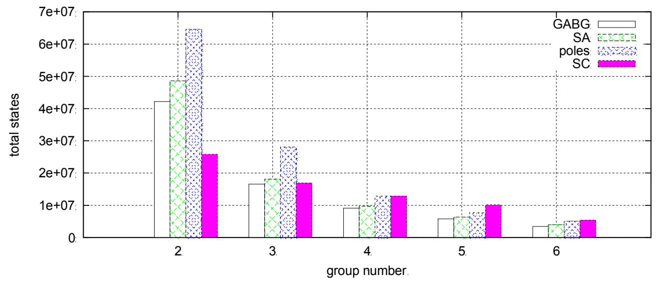

Fig. 1. States number comparison on Snort (120 rules)

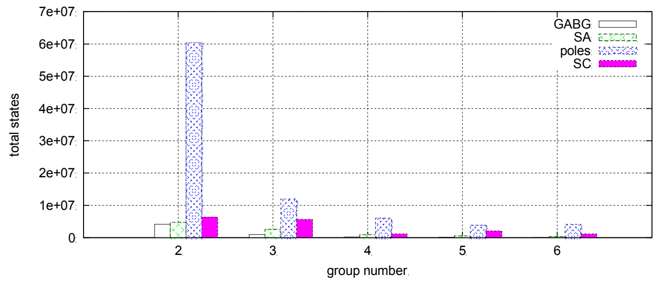

Fig. 2. States number comparison on L7-filter (111 rules)

STEP-3: Building new matrix

k smallest eigenvectors are picked out from matrix L, and then we arrange these *k* eigenvectors as a *N* by *k* matrix.

STEP-4: K-means clustering

Treat each row of the new matrix in *STEP-3* as a vector in *k*-dimensional space, and use K-means to cluster.

3. EVALUATION

Experiments are conducted on an Intel(R) Xeon(R) E5335 platform (CPU: 2.0GHz, Memory: 4GB, OS: Ubuntu 12.04 64bit). The Regular Expression Processor [5] is used to calculate the number of states of a regular expressions distribution, as the metric of memory consumption. Two rule sets picked from open source software Snort [6] and L7-filter [7] are used.

We implement Simulated Annealing (SA) [2], Poles heuristic (poles) algorithms [2] and Grouping Algorithms Based on Gene (GABG) [4] as comparisons.

In Fig. 1 and Fig. 2, the x axis represents the number of group (which is set to 2~6), and the y axis represents the total DFA number of all groups under each group number. The results show that Spectral Clustering grouping algorithm (SC) outperforms poles but is not as space-efficient as GABG and SA.

However, since both of GABA and SA are heuristic algorithms which need lots of iterations to converge to an optimum solution, it is not quite time-efficient to obtain such a result in a long run time. For the problem of distributing 111 regular expressions into 4 groups with given 50 individuals and 5000 generations in GABG, it takes nearly 2 hours to obtain a final solution. In the meantime, SC only needs around 5 seconds to finish the matrix decomposition. Figure 3 illustrates the relationship between time-consumption and space-efficiency of each grouping algorithms. As we can see, the Spectral Clustering grouping algorithm is a better trade-off between time and space than other algorithms.

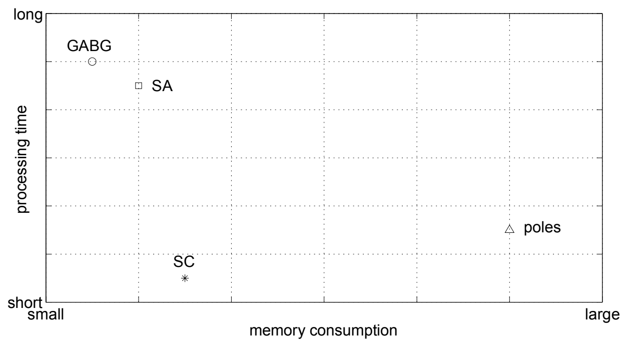

Fig. 3. Time vs. Space

4. CONSLUSION AND FUTURE WORK

In this work, we propose a new regular expression grouping algorithm based on Spectral Clustering, aiming at solving the problem of DFA state explosion. The preliminary experimental results on practical rule set draw the conclusion that Spectral Clustering based regular expression grouping algorithm is a better trade-off between processing time and memory consumption compared to previous methods. Our future work will focus on designing more time and space efficient grouping algorithms. For example, taking the result of SC as a preliminary distribution of regular expressions, which can speed up the convergence of heuristic grouping procedure. Meanwhile more experiments are expected.

5. REFERENCES

[1] F. Yu, Z. Chen, Y. Diao, T. V. Lakshman and R. H. Katz. Fast and Memory-Efficient Regular Expression Matching for Deep Packet Inspection. Proc. of ACM/IEEE ANCS, 2006.

[2] J. Rohrer, K. Atasu, J. V. Lunteren and C. Hagleitne. Memory-Efficient Distribution of Regular Expressions for Fast Deep Packet Inspection. Proc. of the 7th IEEE/ACM international conference on hardware/software codesign and system synthesis, 2009.

[3] R. Antonello, S. Fernandes, A. Santos, et al. Efficient DFA grouping for traffic identification. Proc. of Global Communications Conference (GLOBECOM), 2012.

[4] Z. Fu, K. Wang, L. Cai and J. Li. Intelligent Grouping Algorithms for Regular Expressions in Deep Inspection. Proc. of the 23rd International Conference on Computer Communications and Networks (ICCCN), 2014.

[5] Regular Expression Processor, http://regex.wustl.edu/.

[6] Snort, http://www.snort.org/.

[7] L7-filter, http://l7-filter.clearfoundation.com/.