Frequent Pattern Mining On Un-rooted Unordered Tree Using FRESTM

Dhananjay G. Telavekar[1], Hemant A. Tirmare[2]

[1]M.Tech. Scholar, Dhananjay G. Telavekar, Dept. Of Technology, Shivaji University, Kolhapur, Maharashtra, India
[2]Asst. Professor H. A. Tirmare, Dept. Of Technology, Shivaji University, Kolhapur, Maharashtra, India

---***---

Abstract - *Data mining problem to find frequent restrictedly embedded subtree pattern from a set of unordered un-rooted tree. In this paper we present frequent restrictedly embedded sub tree miner (FRESTM), is an efficient algorithm for mining frequent, unordered, un-rooted, embedded sub-trees in a database of labeled trees. Our contribution is as follows: The algorithm enumerates all embedded, unordered trees. A new equivalence class extension scheme generates all candidate trees and data tree. The notion of scope-list joins is extended to compute the frequency of unordered trees. The performance evaluation on several synthetic and real world data shows that FRESTM is an efficient algorithm*
Key Words: *un-rooted tree, pattern mining, pattern matching, embedded sub-tree, frequent sub-trees.*

1.INTRODUCTION

In recent researches the main problem of finding frequent patterns from a database of graphs has several important applications in different areas like bioinformatics, user web log analysis, semi-structured XML data [12], web mining, RNA, phylogeny [15], prerequisite trees, and chemistry compound data [9].

A main problem in many other data mining tasks such as association rule mining, classification and clustering. Although finding of frequent patterns (for example closed patterns) has found more interest, developing efficient algorithms for finding frequent patterns is still important, because the efficiency of the algorithms of finding condensed representations depends on the efficiency of the frequent pattern mining algorithms.

Whereas item set mining and sequence mining have been studied extensively in the past, recently there has been tremendous interest in mining increasingly complex pattern types such as trees and graphs For example several algorithms for tree mining have been proposed which include TreeMiner [5], which mines embedded, ordered trees, FreqT which mines induced ordered trees, FreeTreeMiner which mines induced, unordered, free trees that is there is no distinct root. TreeFinder which mines embedded, unordered trees (but it may miss some patterns; it is not complete); and PathJoin, uFreqt [14], uNot [18], CMTreeMiner [4], and Hybrid Tree Miner which mine induced, unordered trees. Our focus in this paper is on a complete and efficient algorithm for mining frequent, labeled, unordered, un-rooted, embedded subtrees and graphs. An efficient algorithm is introduced for the problem of mining frequent, unordered, embedded sub trees in a dataset of trees. Our contribution is as follows:

The first algorithm enumerates all embedded, unordered trees.

A new self contained equivalence class extension scheme generates all candidate trees. Only potentially frequent extensions are considered, but some redundancy is allowed in the candidate generation to make each class self contained.

The notion of scope-list joins is extended for fast frequency computation for unordered trees. Performance evaluation is conducted on several synthetic dataset and a real web log dataset to show that FRESTM is an efficient algorithm.

2.RELATED WORK

To provide solution for the above problem there are different methodologies suggested by various authors we are going to discuss some of them.

Zhang and Wang [3] puts forth supporting structure for a frequent agreement subtrees problem for both rooted and un-rooted phylogenetic trees using different data mining methods. Describe canonical form for rooted trees and phylogeny-aware tree expansion scheme for creating candidate subtrees level by level. To find all frequent agreement subtrees in a given set of rooted trees, using Apriori-like approach.

Chehreghani introduces OInduced [2] to mine frequent ordered induced tree patterns. It uses breadth first candidate generation method. first, log data is converted into rooted ordered trees, and a set of frequent patterns are extracted, based on these patterns, a structural classifier is built to classify different users. Structural classifiers show higher performance compared to traditional classifiers.

Genetic algorithm [6]explain optimization procedure for structural pattern recognition in a model-based recognition system using attributed relational graph matching methods. To improve the GA-based attributed relational graph matching plan to major and faster convergence rate and good quality mapping between a scene attributed relational graph and and model attributed relational graph.

Leung and Suen [7] Elaborated top-down elastic approach to pattern matching and its application to complex handwritten Chinese character recognition is discussed.

Zhihui [1] propose different methods to discover restrictedly embedded subtree patterns. We learn properties of a canonical form of unordered trees, Apriori-based methods are elaborated to generate all candidate subtrees using two methods 1) pairwise joining 2) leg attachment. Calculating the restricted edit distance between a candidate subtree and a data tree restrictedly embedded subtree can be achieved. These methods are combined into an algorithm, named as (FRESTM).

Chi etal [4] proposed CMTreeMiner, that determine closed and maximal frequent sub-trees in a database of labeled rooted trees, where the rooted trees can be either ordered or unordered. It mines both closed and maximal frequent sub-trees by traversing and calculate tree that consistently calculate all frequent sub-trees.

Zaki [5] present a tree mining example the problem of mining structural patterns in a data set of Ribonucleic acid (RNA) molecules, can be represented as trees. The knowledge about a newly sequenced RNA, researchers are looking for common topological patterns, and can give main hints to the function of the RNA.

POTMiner [11] proposed A scalable and parallelizable algorithm to mine partially-ordered trees. It can identify both induced and embedded subtrees. It can also handle both completely ordered and completely unordered trees.

Wang [9] presents Approximate-Tree-By-Example (ATBE), which allows incorrect identical trees. ATBE system interacts with the user through a easy but influential query language graphical devices are provided to smooth progress of inputing the queries.

Shasha and his team [8] Presents an efficient and several heuristics leading to approximate solutions. To the probabilistic hill climbing and bipartite matching techniques.

HybridTreeMiner [10] Introduce frequently occurring sub-trees in a database of rooted unordered trees. It mines frequent sub-trees by traversing an enumeration tree that consistently calculate all sub-trees. Calculated tree is defined based on a new canonical form for rooted unordered trees.

3.PROPOSED SYSTEM

The proposed system consists of five modules:

Module 1-Preprocessing and Tree Geneneration:

In preprocessing web log defining will be done this include removing incomplete web log, reducing noisy data and data set conversion. Tree generation will convert session web logs to tree structure the session web logs are in form of associated manner.

Module 2-Canonical Representation:

In this the generated tree will be converted into unique representation, from a previous from that had more than one possible representation. An unordered tree *t* is in its canonical form if no equivalent ordered tree t' exists with depth label sequence of (t') should be less than depth label sequence of (t), that is the canonical form of an unordered tree should result in the lightest depth label sequence among all of its equivalent ordered trees. Directly removing the last node of a canonicalized tree 't' will result in a residue tree still in its canonical form. Here directly removing means removing a node without further canonicalizing the resulting tree. Therefore, if 't' is an unordered tree in its canonical form, then every downward sub-tree and every prefix of 't' is also automatically in its canonical form.

Module 3-Support Counting:

To count the support is to calculate the occurrence number, of a candidate k-subtree pattern in the whole data set. We should run the restrictedly embedding detection subroutine on the candidate pattern tree against all data trees one by one.

Module 4-FRESTM (Frequent restrictedly embedded subtree mining):

Apriori-based data mining method, which moderately enumerates all candidate subtrees from a given set of unordered trees, level by level, using the rightmost expansion methods. At the start frequent 1-subtrees and 2-subtrees are discovered. To mention all frequent 1-subtrees, one by one that is frequent single labels, we traverse every node of every tree to create an inverted index structure for each unique label appearing in the trees. Specifically, for each unique label, we maintain a list of IDs of supporting trees, in which the label appears. By comparing IDs of supporting trees with the given minsup, we can decide whether the label is frequent or not.

Module 5-Generate Subtree:

The generated subtree to grow frequent subtrees level by level through pairwise joining and leg attachment methods. when frequent subtrees of size k reaches zero, no more frequent (k +1)-subtrees can generated and hence the discovering process terminates. Frequent subtrees of size k can be as small one to allow self-joining and leg attachment operations.

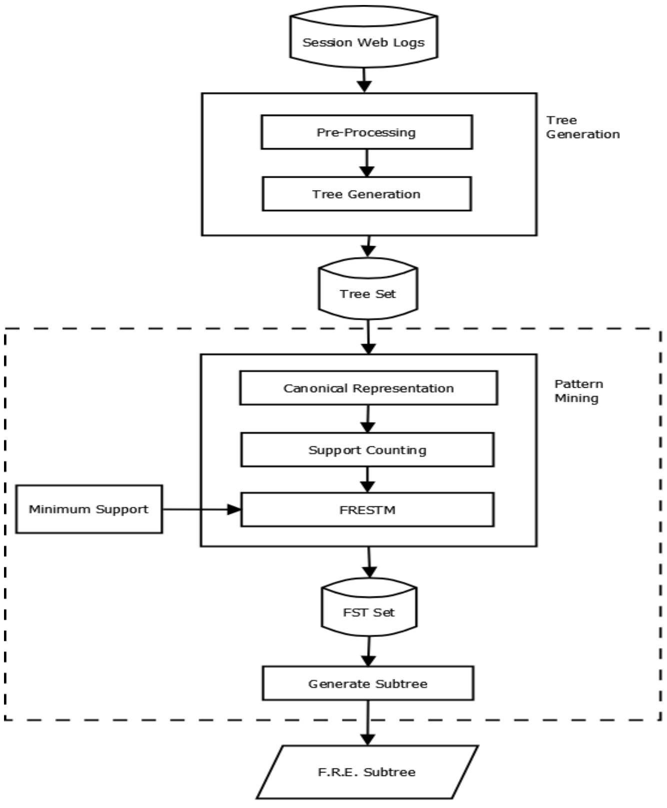

Fig. 1. Proposed System Model

4.EXPERIMENTAL RESULTS

The running time of FRESTM on the datasets. It can be seen from the figure 2 that the time needed by FRESTM. Scales up linearly with respect to the dataset size. This happens because the more trees a dataset has the more time is needed for calculating occurrence number of candidate sub-tree in the dataset.

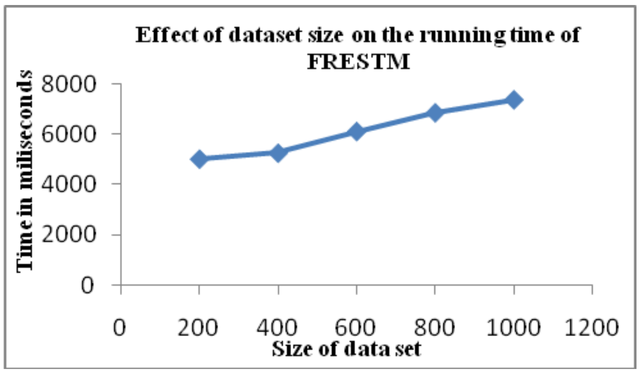

Fig. 2. Effect of dataset size on the running time of FRESTM

With small minimum support value many long patterns with different labels were discovered by our algorithm. As a consequence, much time was spent in finding these long patterns. On the other hand, with a large minimum support value the running time of our algorithm decreased as few patterns qualified to be solution. As shown in figure3.

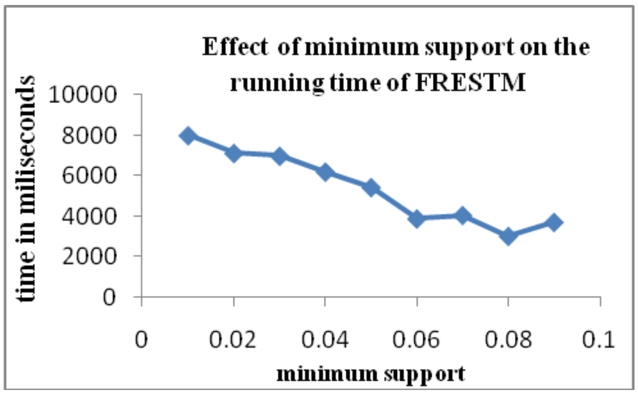

Fig. 3. Effect of minimum support on the running time of FRESTM

The number of frequent patterns detected by the algorithm decreases as the minimum support value increases that is value of minimum support increases the sub-tree generated gets decrease. As shown in figure 4

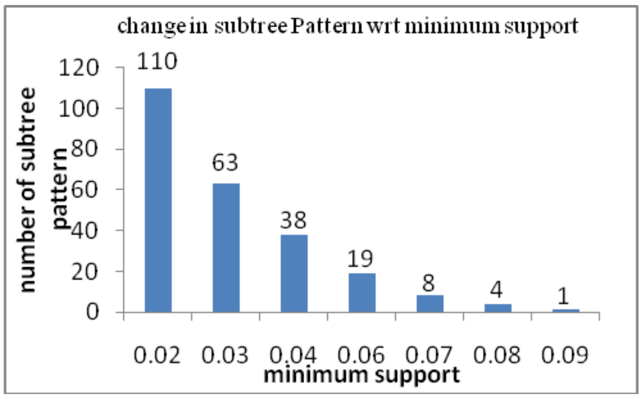

Fig. 4. Effect on minimum support on the number of frequent Patterns

5. CONCLUSION

There are many different tree mining algorithms that work either on ordered or unordered trees but in this paper, we formalize a restrictedly embedded subtree mining problem, which has applications in many domains where data can be represented as unrooted labeled unordered trees. We learn the properties of the canonical form of unordered trees and propose new tree expansion techniques that can correctly, and efficiently generate all candidate subtrees. Then, we introduce a restricted edit distance based technique to detect the restrictedly embedding relationship between a pattern tree and a data tree. We design an Apriori based algorithm, FRESTM, to answer the tree mining problem. To the best of our knowledge, this is the first algorithm for finding restrictedly embedded subtree patterns in multiple Un-rooted unordered trees. Experimental result on real world data set gives good performance of our system.

ACKNOWLEDGMENT

Our sincere thanks to Department of Technology, Shivaji University, Kolhapur for supporting this work.

REFERENCES

[1] Sen Zhang, Zhihui Du, and Jason T. L. Wang, “New Techniques for Mining Frequent Patterns in Unordered Trees,” IEEE TRANSACTIONS ON CYBERNETICS, VOL. 45, NO. 6 , pp. 1113–1125, JUNE 2015.
[2] M. H. Chehreghani, C. Lucas, and M. Rahgozar, “OInduced: An efficient algorithm for mining induced patterns from rooted ordered trees,” *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 41, no. 5, pp. 1013–1025, Sep. 2011.
[3] S. Zhang and J. T. L. Wang, “Discovering frequent agreement subtrees from phylogenetic data,” *IEEE Trans. Knowl. Data Eng.*, vol. 20, no. 1, pp. 68–82, Jan. 2008.
[4] Y. Chi, Y. Xia, Y. Yang, and R. R. Muntz, “Mining closed and maximal frequent subtrees from databases of labeled rooted trees,” *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 2, pp. 190–202, Feb. 2005.
5] M. J. Zaki, “Efficiently mining frequent trees in a forest: Algorithms and applications,” *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 8, pp. 1021–1035,Aug. 2005.
[6] K. G. Khoo and P. N. Suganthan, “Structural pattern recognition using genetic algorithms with specialized operators,” *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 33, no. 1, pp. 156–165, Feb. 2003.
[7] C. H. Leung and C. Y. Suen, “Matching of complex patterns by energy minimization,” *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 28, no. 5, pp. 712–720, Oct. 1998.
[8] D. Shasha, J. T. L. Wang, K. Zhang, and F. Y. Shih, “Exact and approximate algorithms for unordered tree matching,” *IEEE Trans. Syst., Man, Cybern.*, vol. 24, no. 4, pp. 668–678, Apr. 1994.
[9] J. T. L. Wang, K. Zhang, K. Jeong, and D. Shasha, “A system for approximate tree matching,” *IEEE Trans. Knowl. Data Eng.*, vol. 6, no. 4, pp. 559–571, Aug. 1994.
[10] Y. Chi, Y. Yang, and R. R. Muntz, “HybridTreeMiner: An efficient algorithm for mining frequent rooted trees and free trees using canonical forms,” in *Proc. 16th Int. Conf. Sci. Statist. Datab. Manage.*, Santorini Island, Greece, Jun. 2004.
[11] A. Jiménez, F. Berzal, and J. Cubero, “POTMiner: Mining ordered, unordered, and partially-ordered trees,” *Knowl. Inf. Syst.*, vol. 23, no. 2, May 2010, pp. 199–224.
[12] M. L. Lee, L. H. Yang, W. Hsu, and X. Yang, “XClust: Clustering XML schemas for effective integration,” in *Proc. 11th ACM Int. Conf. Inf. Knowl. Manage.*, McLean, VI, USA, Nov. 2002.
[13] L. Liu and J. Liu, “Mining frequent embedded subtree from tree-like databases,” in *Proc. Int. Conf. Internet Comput. Inf. Serv.*, Hong Kong, Sep. 2011.
[14] S. Nijssen and J. N. Kok, “Efficient discovery of frequent unordered trees,” in *Proc. 1st Int. Workshop Mining Graphs, Trees, Sequences (MGTS)*, 2003.
[15] D. Shasha, J. T. L. Wang, and S. Zhang, “Unordered tree mining with applications to phylogeny,” in *Proc. IEEE Int. Conf. Data Eng.*, Boston, MA, USA, pp. 708–719, 2004.
[16] L. Zou, Y. Lu, H. Zhang, R. Hu, and C. Zhou, “Mining frequent induced subtree patterns with subtree-constraint,” in *Proc. 6th IEEE Int. Conf. Data Mining (ICDM) Workshop*, Hong Kong, Dec. 2006.
[17] T. Asai, H. Arimura, T. Uno, and S. Nakano, “Discovering frequent substructures in large unordered trees,” in *Proc. 6th Int. Conf. Discov. Sci.*, Sapporo, Japan, Oct. 2003.

Mr. Dhananjay G. Telavekar had completed B. Tech in Computer Science and Technology in 2014 from Department of Technology ,Kolhapur. Currently he is pursuing M. Tech degree in computer science and Technology from Department of Technology, Kolhapur.

Mr. H. A. Tirmare perceived B.E. and M. E in computer science and Engineering. Presently he is working as Asst. Professor in Department of Technology, Shivaji University, Kolhapur. His interest lies in Application Programming, Data Mining and Web Technology.