Taxonomic Superimposed Tree and Graph Mining Algorithms: A Comprehensive Study

Saed Khawaldeh
Erasmus+ Joint Master Program in Medical Imaging and Applications; University of Burgundy (FR), UNICLAM (IT) and University of Girona (ES)

Usama Pervaiz
Erasmus+ Joint Master Program in Medical Imaging and Applications; University of Burgundy (FR), UNICLAM (IT) and University of Girona (ES)

Yeman B. Hagos
Erasmus+ Joint Master Program in Medical Imaging and Applications; University of Burgundy (FR), UNICLAM (IT) and University of Girona (ES)

Tajwar A. Aleef
Erasmus+ Joint Master Program in Medical Imaging and Applications; University of Burgundy (FR), UNICLAM (IT) and University of Girona (ES)

Vu Hoang Minh
Erasmus+ Joint Master Program in Medical Imaging and Applications; University of Burgundy (FR), UNICLAM (IT) and University of Girona (ES)

ABSTRACT
Data mining is one of the most popular research topics nowadays. It has a lot of applications in many fields such as bioinformatics, social networks, XML processing, web usage mining and computer networks. To the best of our knowledge, taxonomic subtree mining in tree dataset and taxonomic subgraph mining in single graph dataset are problems which have not been studied before. On the contrary, taxonomic subgraph mining for graph transaction dataset has been discussed and presented in many papers in the literature. In general, subtree and subgraph mining algorithms are divided into two types: apprior-based approach algorithms and pattern-growth approach algorithms. Moreover, each frequent subtree and subgraph mining algorithm should include two steps; candidate generation and support counting. Our goal in this paper is to present a summary about the available tree and graph mining algorithms which have been discussed in the literature, also, to propose a taxonomic superimposed tree and graph mining algorithms inspired by the taxonomy-superimposed graph mining concepts. The proposals that we present in this paper can be used for mining biological tree and graph datasets to find frequent subtree and subgraph patterns.

Keywords
Datamining, Taxonomy, Subtree Mining, Subgraph Mining, Frequent Patterns.

1. INTRODUCTION
The importance of finding the frequent patterns in the different types of datasets such as graph, network, or tree datasets has made a lot of researcher interested in creating and developing algorithms for this purpose, especially that this field of research has a wide-range of applications. The particular problem of taxonomic subtree mining in a tree transaction dataset has not been studied before. Likewise, the problem of taxonomic subgraph mining in a single graph dataset has not been addressed before. All the previous work either concentrates on subtree or subgraph mining without paying attention to the taxonomy concept, or concentrates on subgraph mining for graph transaction dataset, so the algorithms we propose at the end of our paper are the first algorithm including the taxonomy concept to solve the two problems; subtree mining for tree transaction dataset and subgraph mining in a single graph dataset.

Tree is a graph where each node has only one parent and it does not have any cycles, so tree is always a graph. This work presents a brief overview about data mining in general and the steps followed in this process. It also shows the different techniques used in graph mining including graph clustering, graph classifications, and subgraph mining. Furthermore, we present a literature review about the problems of subtree and subgraph mining. In addition, we discuss two algorithms we propose for taxonomic subtree and subgraph mining. At the end, we conclude with some possible future work relevant to the problems addressed.

1.1 Datamining
Data Mining is the process of finding patterns representing knowledge from given data structure involving methods have being designed for that purpose, these methods might include artificial intelligence, machine learning, statistics, and database systems [1]. Data mining is very important in both the private and public sectors. Banking, insurance, medicine, and retailing commonly use data mining to reduce costs and increase sales. The process of data mining requires many steps to extract knowledge out of data and determine whether if it is useful or not.

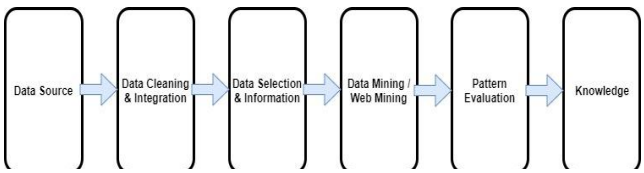

Fig.1: Steps in Datamining process

1.2 Graph Mining Techniques
Graph mining techniques are categorized mainly into three groups [2]:

(1) Graph Clustering: Grouping the nodes of the graph into clusters making sure that there will be as much as possible of edges within the one cluster and as few as possible edges between the different clusters. This technique is based on unsupervised learning technique as the different classes which they will be formed as clusters are not known before clustering,

and the clusters formed based on some similarities between the vertices.

(2) Graph Classification: Classifying the individual graphs in given dataset into two or more classes. It is based on supervised/semi supervised learning which means the classes of the graphs are defined in prior.

(3) Subgraph Mining: Producing set of frequent subgraphs out of graph dataset considering a given threshold which represents the occurrence of these subgraphs in the dataset. Generally speaking, these techniques are used in graph and tree mining.

In this paper, we discuss the third technique by reviewing some of the algorithms proposed in the previous related work, and then propose our proposed algorithms to tackle the problems of taxonomic subtree mining in tree transaction dataset and taxonomic subgraph mining in a single graph dataset.

1.3 Graph Dataset Settings

There are two settings for any graph dataset; graph transaction setting and single graph setting. Graph transaction setting contains a lot of small graphs called transactions and single graph setting contains only one large graph. The support value for the candidate subgraphs in graph transaction setting equals to the number of dataset graphs which the candidate subgraph appears in, while it equals to the number of occurrences in the given large single graph setting dataset [3].

1.4 Subgraph Mining Steps

Solving the problem of subgraph mining can be done in two steps [4]:

(1) Candidate generation to find all the possible subgraph candidates in the graph dataset. The complexity of this step is not high and it does not require a lot of time and high computational power to be performed.

(2) Frequency counting to find the support value for each candidate in order to check whether this candidate is a valid one or not. This checking is done by comparing this support value with a given threshold. The valid subgraphs have support value more than or equal to the threshold value which considered as the minimum support value. This step is expensive computationally.

All algorithms proposed previously in the literature for subtree mining in tree transaction setting dataset only addressed the problem without including the taxonomy concept, so the algorithm of subtree mining which we propose is the first algorithm proposal including the taxonomy concept for the purpose of subtree mining of transaction tree setting. Furthermore, our other algorithm proposal of subgraph mining in a single graph setting is the only one – to the best of our knowledge – which adopts the taxonomy concept for the subgraph mining of a single graph setting dataset.

2. LITERATURE REVIEW

Subgraph mining and subtree mining problems are slightly similar problems to each other, the reason of similarity is because the data structure for both graphs and trees are almost the same with only few minor differences. The algorithms used for graph mining can be modified with small efforts to be used for tree mining, and the same applied on tree mining algorithms. In this section, we present a literature review for both subtree mining and subgraph mining as there are a lot of overlapping between the two problems.

2.1 Subgraph Mining

In [3], the problem addressed is mining frequent subgraphs in single large graph setting. The author proposed two algorithms; HSIGRAM algorithm which uses the horizontal approach that relies on the Breadth First Search (BDF) to find the frequent patterns, and VSIGRAM which uses the vertical approach based on the Depth First Search (DFS) to find the frequent subgraphs. The results show that VSIGRAM's running time is smaller than HSIGRAM's one.

In [5], a novel algorithm is proposed to solve the frequent subgraph mining problem. Their FFSM algorithm uses the vertical approach to search within an algebraic graph framework has been developed to reduce the generated redundant subgraph candidates. The proposed algorithm avoids the isomorphism testing by creating embedding set for each candidate. The results presented in their work show that FSM handle the subgraph isomorphism problem by introducing FSM join and FSM extension operations, by this procedure which they followed, the number of over-generalized patterns has been reduced.

In [6], gSpan is a novel algorithm solves the frequent graph mining problem without generating subgraph candidates. This algorithm uses DFS strategy to find the frequent subgraphs. The gSpan algorithm found all frequent subgraphs from chemical compound dataset they used in 10 seconds, while another algorithm called FSG; proposed in [7], did the same job in 10 minutes. The gSpan introduces DFS lexicographic order and minimum DFS code techniques to solve the subgraph mining problem using DFS approach. The result show that gSpan outperformed the FSG in [7] regarding the running time for both types of data; synthetic data and chemical compound data.

In [8] and [9], the frequent patterns mining algorithm proposed is mainly for biological data over a taxonomy. The problem solved in their work is not well-studied problem as only few number of research projects followed the same approach as there were not any available biological datasets in the past. The authors proposed a new method to discover the pathways in biological graphs using the Gene Ontology (GO)-based functionalists of enzymes. Those two papers adopt the principle of finding the frequent pathways in transaction graph setting dataset over a taxonomy. The algorithms designed in their work, do not only generate the candidate subgraphs, but also eliminate the over-generalized patterns to make sure that the set of generated subgraphs is optimal and complete.

2.2 Subtree Mining

In [10], data mining algorithm is proposed to explore the tree dataset which called as hierarchal tree structure or tree-like pattern. Counting the frequency of the tree pattern was done using the dynamic programing approach because it efficiently counts the weighted support values. Their proposed algorithm can be applied for mining tree patterns from large dataset with a sequence. The first step in the algorithm was finding a relationship between the data in the structure. The pattern extracted has either a strong ordering from the root to the leaf which is called fully ordering, or it has a tree with root and more children for some nodes which is called partially ordering. For counting the frequency of the pattern extracted they proposed an efficient algorithm for this purpose. The proposed algorithm was implemented into an e-learning system, then it was demonstrated using simulation. Worth to

be mentioned, the proposed algorithm had polynomial time complexity, O (nk), where n and k are the lengths of both the sequence and the matched k-trees, respectively.

In [11], the problem addressed in this paper is mining frequent embedded unordered tree dataset by using an algorithm which generates all the embedded, unordered trees out of the given dataset, afterwards, authors proposed a class extension scheme to generate all the candidate trees. At the end, they added a scope-list notion which made counting the frequency faster. During the process of generating the candidate trees, the algorithm makes sure that the set formed will be non-redundant. Also, it ensures that the candidates' generated support values are more than the support threshold assigned. On the other hand, in canonical extension algorithm, non-redundant candidate is generated, but unfortunately they might not be frequent tree patterns. The frequency computations obtained by doing some tests such as: descendent test and cousin test. The results of implementing the suggested algorithm show that applying the algorithm above generates more unordered patterns than the ordered ones.

In [12], the problem discussed in this paper is about frequent sequential tree mining from a large tree sequence database. Basically, to solve this problem the sequential tree mining problem is converted to normal tree mining problem. Then a transformation for the database is performed to generate sequence trees out of the database given. After that, they perform mining for the sequences from the databases applying conventional sequence mining technique. And lastly, they transform the mined sequences to sequence patterns. After doing the experiments, they noticed that most of the computation time is consumed by database transforming and sequence mining steps. Another outcome observed, the time consumed by the mining system is linearly proportional to the dataset size. Moreover, thy observed that the number of sequences mined decreases with increasing the support threshold value.

In [13], the problem discussed is finding frequent ordered induced tree patterns out of tree dataset. In the algorithm used for solving this problem, they used frequency counting using tree encoding. They also used Breadth First Search (BFS) for candidate generation. Part of the algorithm presented in their paper is defining a new efficient data structure to save the information extracted from frequency counting. The authors additionally used the canonical representation as part of the algorithm as well, and the candidates generated were the only ones in the canonical form. However, the method of frequency counting was new as it is based on the tree encodings; M-Coding and Cm-Coding. The results obtained after experimentation show that the running time of the IOnduced algorithm is less than the time consumed by the Freq and the IMB3Miner algorithms.

In [14], the problem addressed is mining frequent patterns from biological data. This problem has two challenges; the first one is about the biological data itself as it is very huge and complicated, so finding the frequent patterns would very expensive computationally. The second challenge related to applying the current algorithms on this biological data, which is very hard and complicated comparing to applying them on the other regular data structures, so the current algorithms are not suitable for biological data structure. In the approach used in this work, string encoding is applied for the representation of the tree, while the scope-list is applied for extending the substring. Intensive Rooted Tree Mining Algorithm (IRTM) used in this paper, this algorithm first calculates the scope-lists according to the input database, then it represents the trees in string encoding method. The approach is mainly computing all the non-redundant candidate subtrees and their supports, after that, the generated subtrees are generated from the subtree with least node numbers by extending it to make sure that it will not generate the same subtree more than once. To be able to apply their algorithm on RNA data, they converted it to tree-like patterns using a method was designed for this purpose. The results show that IRTM algorithm is better than TREEMINER algorithm for large scale data, and it works faster when the support threshold increases.

In [15], the problem which authors working on is tree mining for the databases which has unlabeled induced rooted trees change over time. This paper does not pay attention to the speed or nature of change, but rather focuses specifically on tree mining for the unlabeled web link structure. The approach used to address the problem in this work relying much on the mathematics. More specifically, they use the method of incremental closed pattern mining. Accordingly, the closed pattern set is calculated whenever a pattern arrives, and then it adds the calculated pattern to the Closed-Subpattern-Mining-ADD list. They used the Closed-Subpattern-Mining-Delete algorithm by deleting one transaction per time to make sure that they will not pre-compute the frequent closed patterns again and again. More specifically, they are using sliding window with the ADWIN estimator which decides on the size of the window. Each tree from the database represents using natural representation which is a sequence over a county infinite alphabet, namely, the set of natural numbers. The results of applying this algorithm on the website browsing tree dataset which was obtained using Zaki Software, show that it works better than the existing algorithms especially when the dataset has high number of trees (around 4.4 million trees for the dataset they used).

In [16], the problem authors solving in this paper is looking for subtrees of n elements in a tree dataset which represent information in a web page. Thus, their work discusses extracting information out of organized dataset of trees. More specifically, their algorithm looks for the frequent trees which represent the Document Object Model (DOM) out of web applications. One of the motivation for this research project is calculating the Restricted Top Down Metric (RTDM) distances for the input trees.

In [17], authors here solve the problem of frequent subtree mining. The input in this work is database of rooted trees. The algorithm designed can be used to solve labeled, unlabeled, ordered, unordered or edge label trees, but these features are provided under some restrictions related to the sequential encoding of the database trees which are represented on an array based representation of trees. The methodology used here generates non-redundant candidate subtrees, moreover, the algorithm can be applied on other types of trees. This algorithm is from the type of pattern-growth approach. Firstly, it transforms the database to sequences, and then into the candidate subtrees with support more than threshold used to generate bigger subtrees. In this algorithm, candidate generation and support counting happens at the same time which reduces the time complexity of the algorithm.

3. PROBLEM DEFINITION

Computing the support differs from database setting to another; for example in transaction tree setting support equals to number of trees where the candidate subtree occurs in over the total number of trees in the given dataset. Fig.2 below represents tree dataset. Traditional frequent subtree mining

algorithm would only find the common two nodes between the two trees; Helicase and DNA Helicase.

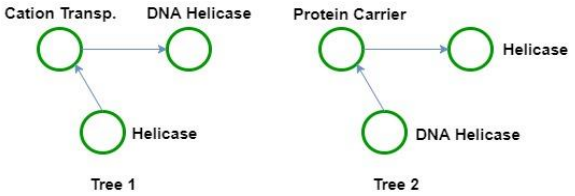

Fig.2: Tree Database

By giving the Gene Ontology (GO) subgraph shown below in Fig.3, and by applying taxonomic-superimposed tree mining algorithm, we can efficiently discover the frequent tree structure in a database of taxonomy-superimposed trees, see Fig.4. The same thing applied on graphs dataset.

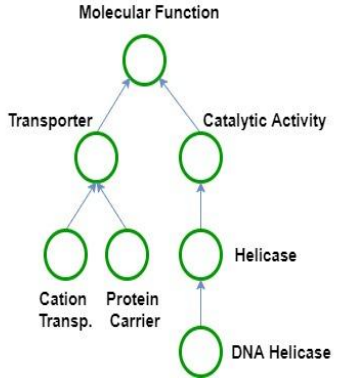

Fig.3: A subgraph of GO

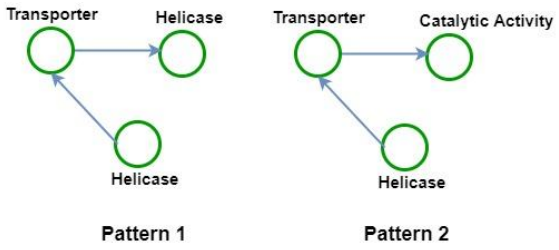

Fig.4: Sample Patterns obtained

Having a large number of patterns is one main challenge in subtree and subgraph mining problems. This occurs because the number of generated patterns depends on the average depth and the number of nodes. Over-generalized patterns is the other main challenge in subtree and subgraph mining problem; a pattern A is an over-generalized pattern in a patterns set if there is another pattern more specialized than A and have the same support as A. Thus, generating a minimal and optimal set of frequent patterns given a tree or a graph datasets and their GO are the problems which this work discuss.

4. METHODOLOGY

Our proposed algorithm produces a set of frequent subtrees or subgraphs out of a given tree or graph database taking into consideration a given threshold represents the minimum support value for the valid mined subtrees or subgraphs and a taxonomy represents the concepts and relationships between the different tree or graph nodes. The algorithm first generates candidate subtrees or subgraphs starting from one edge between two nodes with their support values. While that, the algorithm eliminates the non-valid subtrees or subgraphs based on their support values. To reduce the time complexity and eliminate any over-generalized patterns, it generates the support values simultaneously with candidate generation process. Mainly, the algorithm mines the frequent subtrees or subgraphs in three steps:

(1) Relabel vertices in the input tree or graph databases with the most general ancestor of original label. An example is shown in Fig.6.

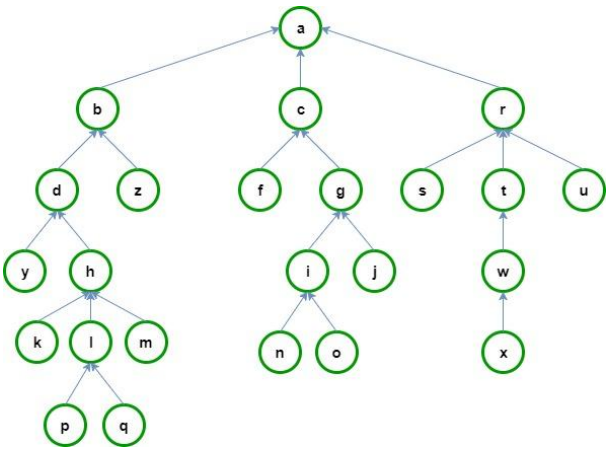

Fig.5: Sample Taxonomy

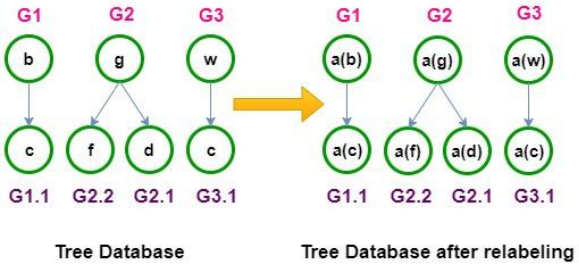

Fig.6: Relabeling

(2) Mine for classes of patterns, extend general-purpose tree or graph mining, create taxonomy-projected occurrence indices and produce pattern classes.

(3) Specialized pattern enumeration shown in Fig.7. This step used to enumerate member's pattern classes and compute support using occurrence indices.

Another approach for solving the problem is by using another algorithm. The algorithm mines the frequent subtrees or subgraphs by split each tree-string or graph-sting into their suffixes, insert suffixes into a Generalized Suffix Graph (GSG), record suffix id`s in GSG, and finally mine for frequent substrings in GSG. Starting with building the Canonical String Representation Scheme for trees or graphs as shown in Fig.8.

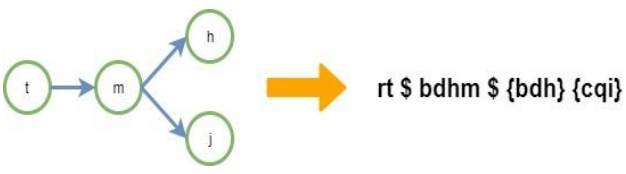

Fig.8: Canonical String Representation Scheme for a tree

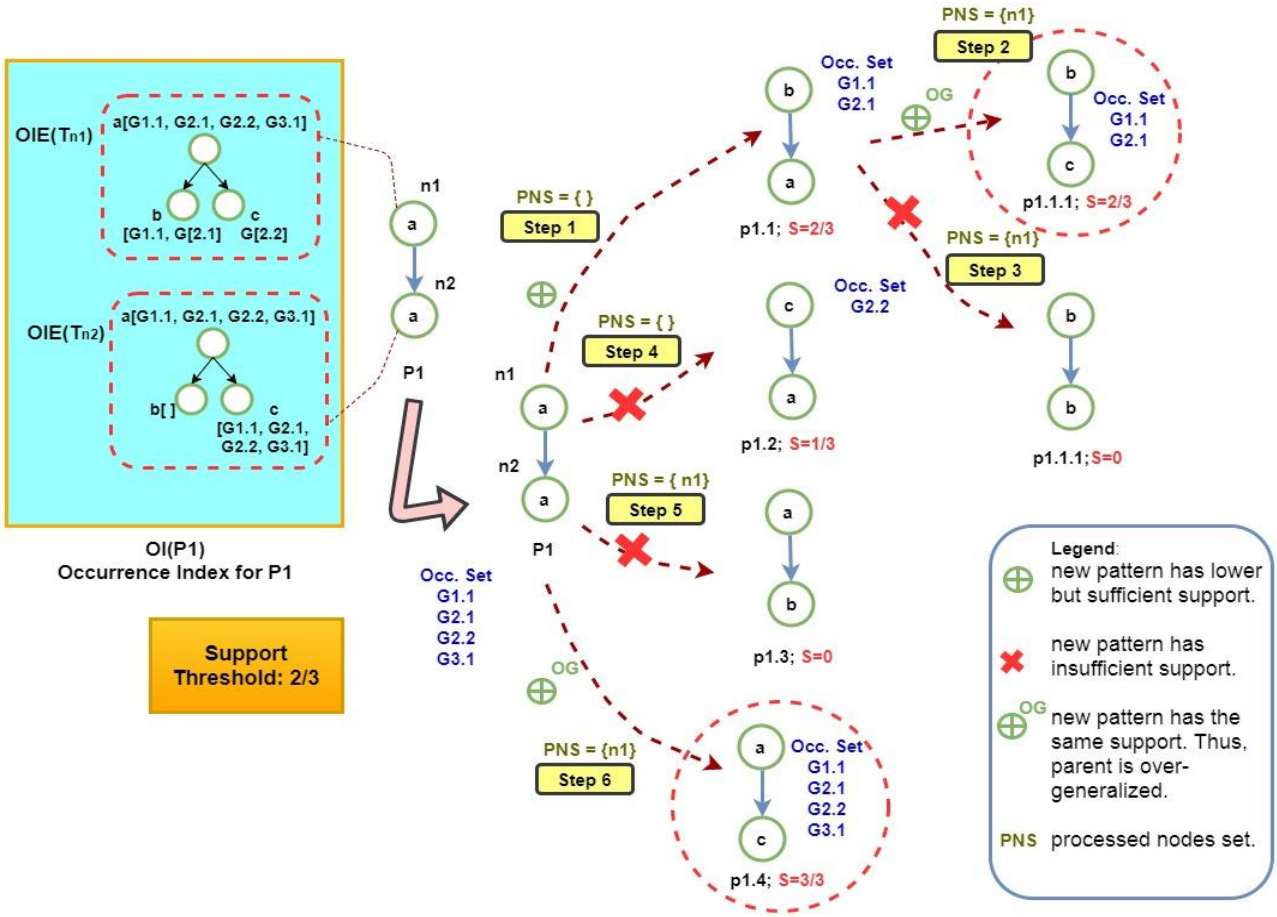

Fig.7: Enumerating Specialized Patterns

After this step, each tree-string splits into its suffixes as shown in Fig.9.

S1: {bdh} {cgj} S2: bdhm $ {bdh} {cgj} S3: rt $ bdhm $ {bdh} {cgj}	4.1 bdh 4.2 cgj 4.3 bdhm $ {pdh} {cgj} 4.4 rt $ bdhm $ {bdh} {cgj}
(a)	(b)

Fig.9: Tree-String Suffixes

Then inserting tree or graph suffixes into the format which is used as input for the mining algorithm, shown in Fig.10.

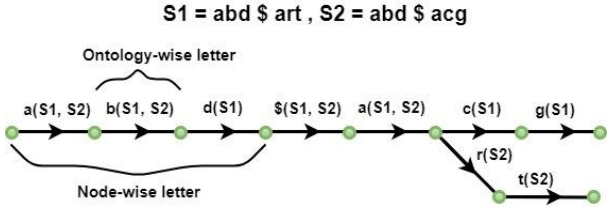

Fig.10: GSG Representation

After that, the algorithm used has two steps:

(1) Candidate generation for extension as if the edge E in a GSG G is given, this step looks for the candidate edge (edges which follow E in G).

(2) Expansion with candidates by choose an edge E from candidate edges, then expand with E to construct a larger subtree, after that compute the support directly from the suffix sets of the edges, and finally insert E into visited edges.

The output of the algorithm above will be converted again to CAN representation then back to the tree or graph patterns representation as shown in Fig.12.

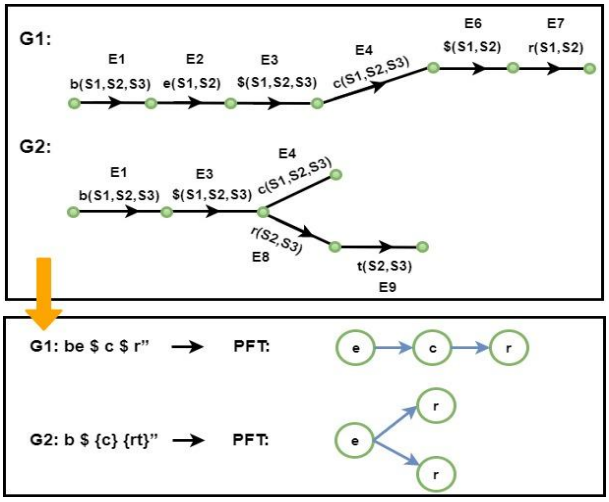

Fig.12: Mapping frequent sub GSGs back to tree patterns

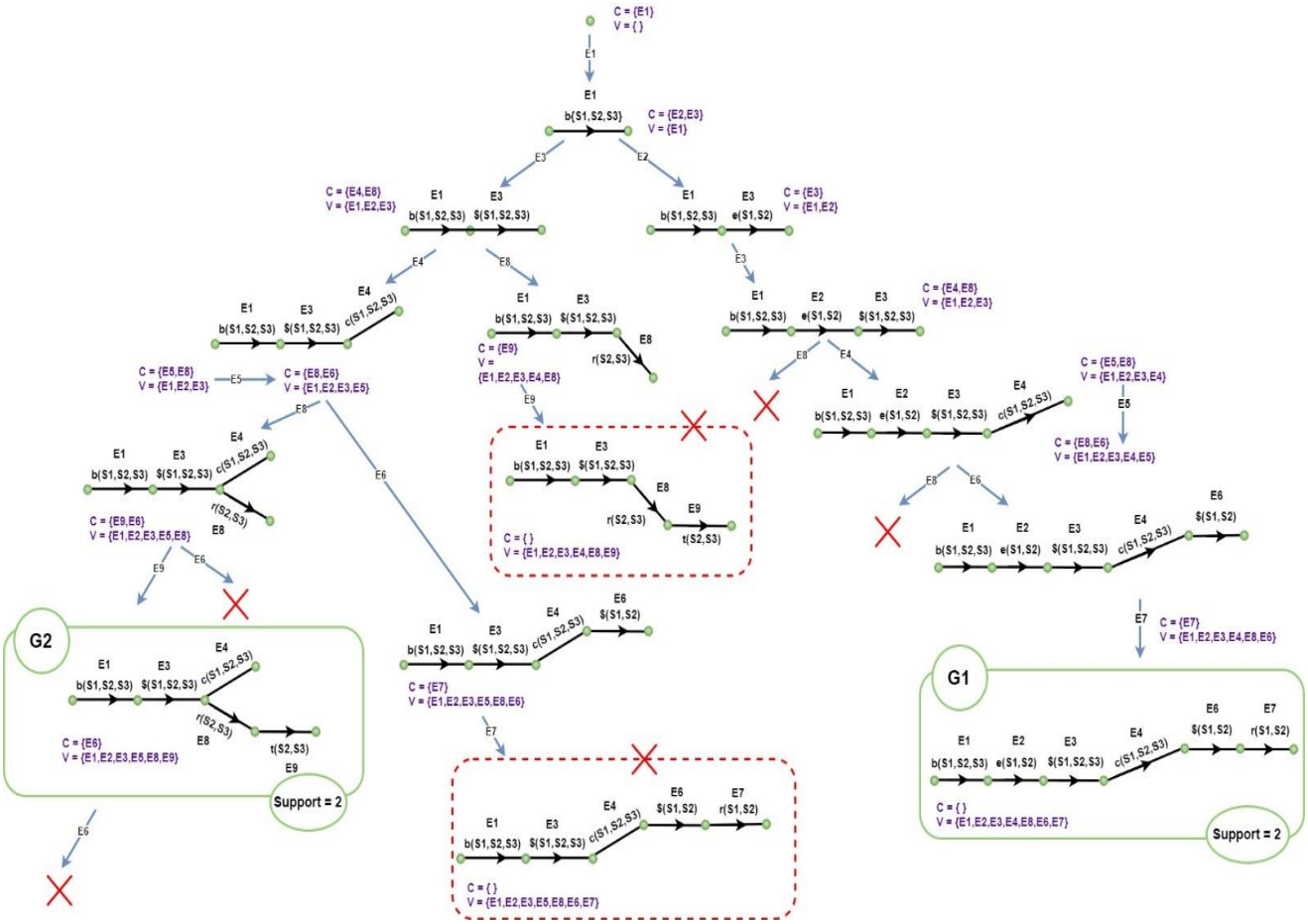

Fig.11: Mining Frequent Subtree Patterns on a GSG

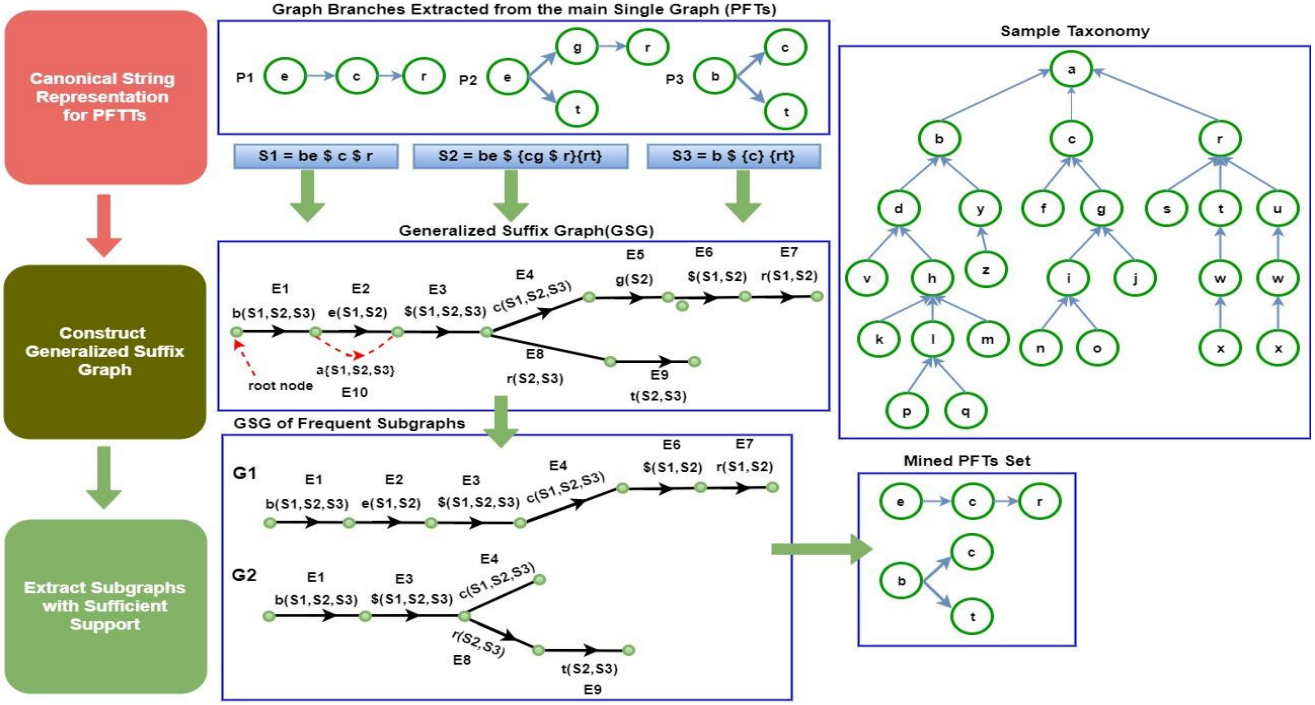

Fig.13: Frequent Subtree Pattern Discovery using Proposed Algorithm

5. EXPERIMENTATION AND IMPLEMENTATION

A synthetic tree and graph datasets generated by random tree and graph generator program has been written for that purpose. The datasets generated is based on different taxonomy files has n number of concepts and m number of relationships, the connected generated trees or graphs represent the dataset which we initially ran the algorithms on.

Java Graph Visualizer was used to visualize the dataset. As the formats of the tree and graph datasets and the tree visualizer do not match, modifications in the tree generator program have been done for the purpose of converting the dataset's format to fit graph visualizer format. Fig.14 shows a sample tree visualized using the Jave Graph Visualizer.

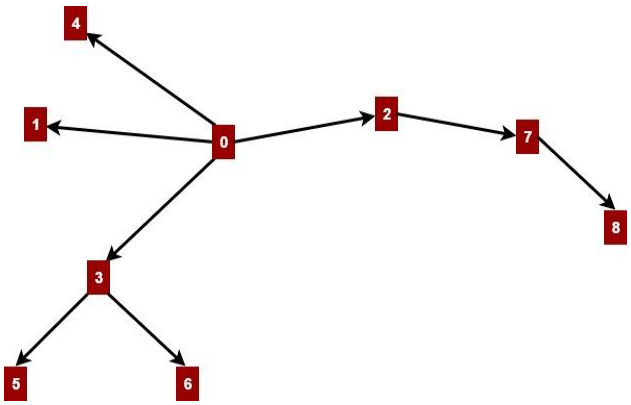

Fig.14: Sample Generated Tree

Our proposal methods are mainly based on the algorithms in [7] and [8]. The expected results of applying the proposed methods on tree or graph datasets with a taxonomy is that they will outperform the other tree and graph mining algorithms because our proposed methods are going to generate an optimal and minimal set of frequent patterns "A set has all the frequent patterns in given tree or graph dataset without including the over-generalized patterns" as our proposed algorithms ensure of the elimination of all the over-generalized patterns at the time of generation as explained in the methodology part above. In addition, the time complexity of our proposed methods would be less than most of other algorithms because in our methods we propose performing the candidate generation and support counting simultaneously which reduce the time consumed to find the frequent patterns. Moreover, all the other algorithms are not including the taxonomy concept while looking for the frequent patterns which cause not generating efficient results when there are constrains on the characteristics of generated patterns such as: the minimum number of nodes in the generated patterns should be at least 2, and this makes the traditional tree and graph mining algorithms not applicable while our proposed algorithms is finding the frequent patterns efficiently. These algorithms would be mostly applied on the biological datasets where we have a GO for the dataset represents the concepts "nodes" in the dataset and the relationships "ancestor and descents" between them. Because of our proposed methodologies and steps, our algorithm will surely outperform all the other algorithms for this kind of tree and graph mining problems.

6. CONCLUSION AND FUTURE WORK

To conclude, this paper presents a proposal of tree and graph mining algorithms over a taxonomy. The paper includes the problem's literature review, definition, methodology proposed, experimentation and implementation setups and expected results. After finishing the literature review we found out that the two specific problems have not been studied before and it will be a contribution to the field if we address them. The next step would be to apply the algorithms on real datasets which represent biological data - tree or graph formats - and run the algorithms on them in order to compare our approach with the other approaches proposed before in other papers.

7. ACKNOWLEDGMENT

This paper is an extended work of semester projects for the Bioinformatics and Data Engineering courses at Istanbul Sehir University. We thank Professor Ali Cakmak and Professor Ahmet Bulut for their guidance and support. In addition, we thank Mohamed Elsharnoby for his help in programming.

8. REFERENCES

[1] Fernando, S. G. S., and S. N. Perera. Empirical Analysis of Data Mining Techniques for Social Network Websites. Compusoft 3.2 (2014): 582.

[2] Rehman, Saif Ur, Asmat Ullah Khan, and Simon Fong. Graph mining: A survey of graph mining techniques. Digital Information Management (ICDIM), 2012 Seventh International Conference on. IEEE, 2012.

[3] Kuramochi, Michihiro, and George Karypis. "Finding frequent patterns in a large sparse graph*." Data mining and knowledge discovery 11.3 (2005): 243-271.

[4] Chehreghani, Mostafa Haghir, and Maurice Bruynooghe. "Mining rooted ordered trees under subtree homeomorphism." Data Mining and Knowl-edge Discovery (2015): 1-24.

[5] Huan, Jun, Wei Wang, and Jan Prins. "Efficient mining of frequent subgraphs in the presence of isomorphism." Data Mining, 2003. ICDM 2003. Third IEEE International Conference on. IEEE, 2003.

[6] Yan, Xifeng, and Jiawei Han. "gspan: Graph-based substructure pattern mining." Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on. IEEE, 2002.

[7] Kuramochi, Michihiro, and George Karypis. "Frequent subgraph discov-ery." Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on. IEEE, 2001.

[8] Cakmak, Ali, and Gultekin Ozsoyoglu. "Mining biological networks for unknown pathways." Bioinformatics 23.20 (2007): 2775-2783.

[9] Cakmak, Ali, and Gultekin Ozsoyoglu. "Taxonomy-superimposed graph mining." Proceedings of the 11th international conference on Extending database technology: Advances in database technology. ACM, 2008.

[10] Chen, Tzung-Shi, and Shih-Chun Hsu. "Mining frequent tree-like pat-terns in large datasets." Data and Knowledge Engineering 62.1 (2007): 65-83.

[11] Zaki, Mohammed J.”Efficiently mining frequent embedded unordered trees.” Fundamenta Informaticae 66.1-2 (2005): 33-52.

[12] Bei, Yijun, et al.”Mining Sequential Trees in a Tree Sequence Database.”International Journal of Database Theory and Application 7.3 (2014): 107-120.

[13] Chehreghani, Mostafa Haghir, et al.”OInduced: An efficient algorithm for mining induced patterns from rooted ordered trees.” Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on 41.5 (2011): 1013-1025.

[14] Liu, Wei, and Ling Chen. ”An Efficient Way of Frequent Embedded Subtree Mining on Biological Data.” Journal of Computers 6.12 (2011): 2574-2581.

[15] Bifet, Albert, and Ricard Gavald. ”Mining adaptively frequent closed unlabeled rooted trees in data streams.” Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2008.

[16] Omer, Barkol, Bergman Ruth, and Golan Shahar. ”A New Frequent Similar Tree Algorithm Motivated by DOM Mining Using RTDM and its new variantSiSTeR.” (2012).

[17] Tatikonda, Shirish, Srinivasan Parthasarathy, and Tahsin Kurc. ”TRIPS and TIDES: new algorithms for tree mining.” Proceedings of the 15th ACM international conference on Information and Knowledge Management.