Correlation Mining in Graph Databases with a New Measure

Md. Samiullah[1], Chowdhury Farhan Ahmed[1], Manziba Akanda Nishi[1], Anna Fariha[1], S M Abdullah[2], and Md. Rafiqul Islam[3]

[1]Department of Computer Science and Engineering, University of Dhaka, Bangladesh
sami_cse_du1@yahoo.com, farhan@khu.ac.kr, nishifriend786@yahoo.com, purpleblueanna@gmail.com

[2]Department of Computer Science and Engineering,
United International University, Bangladesh
smab@cse.uiu.ac.bd

[3] School of Computing and Mathematics, Charles Sturt University, Australia
mislam@csu.edu.au

Abstract. Correlation mining is recognized as one of the most important data mining tasks for its capability to identify underlying dependencies between objects. Nowadays, data mining techniques are increasingly applied to such non-traditional domains, where existing approaches to obtain knowledge from large volume of data cannot be used, as they are not capable to model the requirement of the domains. In particular, the graph modeling based data mining techniques are advantageous in modeling various real life complex scenarios. However, existing graph based data mining techniques cannot efficiently capture actual correlations and behave like a searching algorithm based on user provided query. Eventually, for extracting some very useful knowledge from large amount of spurious patterns, correlation measures are used. Hence, we have focused on correlation mining in graph databases and this paper proposed a new graph correlation measure, *gConfidence*, to efficiently extract useful graph patterns along with a method *CGM* (*C*orrelated *G*raph *M*ining), to find the underlying correlations among graphs in graph databases using the proposed measure. Finally, extensive performance analysis of our scheme proved two times improvement on speed and efficiency in mining correlation compared to existing algorithms.

Keywords: Correlation mining, knowledge discovery, correlated graph patterns, graph mining, graph correlation.

1 Introduction

Data mining extracts useful implicit patterns within data along with important correlations/affinities between the patterns to discover knowledge from databases. Moreover, in data mining, correlation analysis is a special measure which finds the underlying dependencies between objects. The frequent itemset can be mined but correlation calculation among items can define the dependencies and mutual correlation among items within an itemset. The hidden information within databases, and mainly the interesting association relationships among sets of objects, may disclose useful patterns for decision support, financial forecast, marketing policies, even medical diagnosis and many other applications.

Nowadays, data mining techniques are applied to non-traditional domains e.g. the most complicated real life scenarios where objects are interacting with their surrounding other objects. To model such scenarios graph can be used, where vertices of the graph will correspond to entities and edges will correspond to relations among entities. Because of combinatorially explosive search for subgraphs which includes subgraph isomorphism testing, the graph structured data mining is difficult. Moreover, in mining graph data, the emphasis is on frequent labels and common topologies unlike traditional data. Here mining can be divided into level-by-level *generate-and-test* method and *pattern growth-based approach*. AGM[1] and FSG[2] are of the former type, where as gSpan[3] and graph pattern [4] are of later type, which require no candidate generation. In order to discover correlations, several measures are used [5] and to mine correlation in graph databases existing works such as [6] mainly focus on structural similarity search. However, graphs those are structurally dissimilar but always appear together within the database, may be more interesting , such as, the chemical properties of isomers [7].

For capturing effective correlations, authors of [7] proposed CGS algorithm which mines graph correlation by adopting $Pearson's\ correlation\ coefficient$ by taking into account the occurrence distributions of graphs. However, CGS works for searching correlation of a specific query graph with the database. Therefore, it has some limitations in describing inherent correlation among graphs and the domain knowledge is obligatory in using CGS, otherwise lots of queries would be meaningless. These facts motivated us in developing a new measure which can prune a large number of un-correlated graphs and designing an algorithm to efficiently mine graph correlation. Our contributions are: a new graph correlation measure $gConfidence$ to mine inherent correlation in graph databases, pruning a large number of candidates using the downward closure property of the measure and an algorithm CGM (Correlated Graph Mining) to efficiently mine correlation by constructing a hierarchical reduced search space.

Rest of the paper is organized as follows: Section 2 contains our proposed scheme and Section 3 focuses on the performance analysis of our proposed algorithm. Finally, we concluded our work in Section 4.

2 Our Proposed Approach

Correlated graph mining is one of the most important graph mining tasks. So, we have proposed a new measure, $gConfidence$ and a new method, $CGM(Correlated\ Graph\ Mining)$, to search correlation among graphs within a graph database. We have mined correlated graphs by constructing a hierarchical search space using our proposed measure and algorithm.

In graph domains, transactions can be represented by $G = \{V(G), E(G), L(V(G)), L(E(G))\}$ that is the set of vertices, set of edges and set of labels for vertices and edges respectively. Size of a graph varies in various algorithms and can be the number of nodes or edges or disjoint paths. To tackle graph isomorphism problem, various labeling or coding is introduced in many algorithms. For a graph database, GD, with G_s, G_b and G_h are sub-graphs of any graph $G \in GD$, the support of G_s is the ratio among the frequency of G_s's supergraph and total number of transaction graphs. Moreover, confidence of G_b with respect to G_h is the ratio between the joint-occurrence frequency of G_b along with G_h and frequency of the super graph for G_b. The canonical labeling for the normal form representation, X of a graph G can be defined as[3], the minimum DFS code among all possible DFS codes of G . The proposed measure $gConfidence$, is defined as follows where the correlation is defined based on the co-occurrence probability of edges of the graphs within the database.

Definition 1. ***(gConfidence)*** *Given a graph database GD and one of its transaction graph G, then we have defined the $gConfidence$ of G_s, a subgraph of G as*

$$gConfidence(G_s) = \frac{\{No.\,of\,graphs\,G \mid G_s \subseteq G \in GD\}}{max(\{No.\,of\,graphs\,G_i \subseteq G \in GD \mid \forall G_i \subseteq G_s\})} \tag{1}$$

"$max(\{No.\,of\,graphs\,G_i \subseteq G \in GD \mid \forall G_i \subseteq G_s\})$" is the maximum support of any sub-graph of G_s. This prescribed clarity merely entails that $gConfidence$ is the smallest correlation of any graph, G_s and fall within the range [0, 1]. The measure $gConficence$, has numerous crucial properties those made it efficient in mining graph correlation more effectively. These are the *downward closure property*, *null invariance*, *lower bound* marker of correlation and $co-occurrence$ observer.

Lemma 1. *Given a graph database GD and one of its transaction graph G, then we can define the $gConfidence$ of $G_s \subseteq G$ derived in Equation 1 as*

$$gConfidence(G_s) = \frac{\{No.\,of\,Graphs\,G \mid G_s \subseteq G \in GD\}}{max(\{\forall e_i \in E(G_s),\ No.\,of\,graphs\,G_j \mid e_i \in E(G_j),\, G_j \in GD\})} \tag{2}$$

Consider a scenario shown in Figure 1, where two frequent closed graphs found from a set of graphs representing a group of people. Each graph in the set represents friend circle of an individual where nodes represent individuals and edges represent interaction among individual pairs. The circles around closed graphs represent the interaction of a group of people all together. Labels of edges and circles represent the frequencies of the edges and closed graphs respectively. However, in mining the most correlated group, frequency of closed-frequent graphs can't help due a tie i.e. 30 for each. As a consequence, Group G_2 will be suggested as most correlated by our proposed measure. Since, maximum interaction of any pair in G_1 is 100 and in G$_2$ is 60. Therefore, $gConfidence(G_1) = \frac{30}{100} = 0.3 < gConfidence(G_2) = \frac{30}{60} = 0.5$.

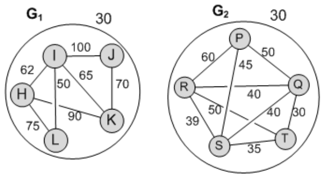

Fig. 1. Motivating Scenario

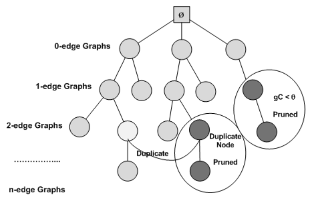

Fig. 2. gConfidence Tree : A search space

Definition 2. ***(Correlated Graph Mining)*** *Given a graph database $GD = \{G_1, G_2, ..., G_N\}$, a user specified minimum support threshold, σ and a user specified minimum correlation threshold, θ. We have to search for interesting graph/subgraphs that is, we have to search for the set of graphs $G_I = \{\forall G_i \mid supp(G_i) \geq \sigma; gConfidence(G_i) \geq \theta\}$. It means the problem is to search for graphs having support count greater than or equal to σ and $gConfidence$ value greater than or equal to θ.*

We have created a hierarchical tree like structure for efficiently searching the correlation among graphs within graph databases. The tree is defined as follows:

Definition 3. ***(gConfidence Tree)*** *A tree, where each node represents a graph or subgraph by storing corresponding DFS code and represents correlation by storing $gConfidence$ value "gC". Moreover, the relation between parent node and child node complies with the relation that a parent is one edge shorter in size than its child and a child is one edge larger than its parent and no child has "gC" greater than its parent. The relation between siblings is consistent with the DFS lexicographic order. That is, the pre-order search of $gConfidence$ tree follows the DFS lexicographic order.*

In Figure 2, we have shown a $gConfidence$ code tree where the $(n+1)$-th level of the tree has nodes which contain DFS codes of n-edge graphs and a value "gC" which represents the inherent correlation among the nodes, edges and subgraphs of that particular graph. Any node in the $gConfidence\ tree$ contains a valid DFS code. Certainly some of the nodes contain a minimum DFS code while others do not. And there could be some nodes having "gC" values smaller than the minimum correlation threshold. The value for "gC" also maintains a parent and child relationship that is $gC(\alpha) \geq gC(\beta)$ where $\alpha = (a_0, a_1, ..., a_m)$ and $\beta = (a_0, a_1, ..., a_m, b)$ that is α is β's parent.

Now we will describe our algorithm for mining graph correlation using our proposed measure $gConfidence$ within graph databases. Since correlation is searched based on user specified minimum correlation threshold then the search space can be pruned based on two values, one for minimum support threshold and another one is minimum confidence threshold. Proposed CGM is an "edge" based correlation mining algorithm and we also used the concept of "Projected Database" to reduce the costly searching operation for counting occurrences of any graph/subgraph.

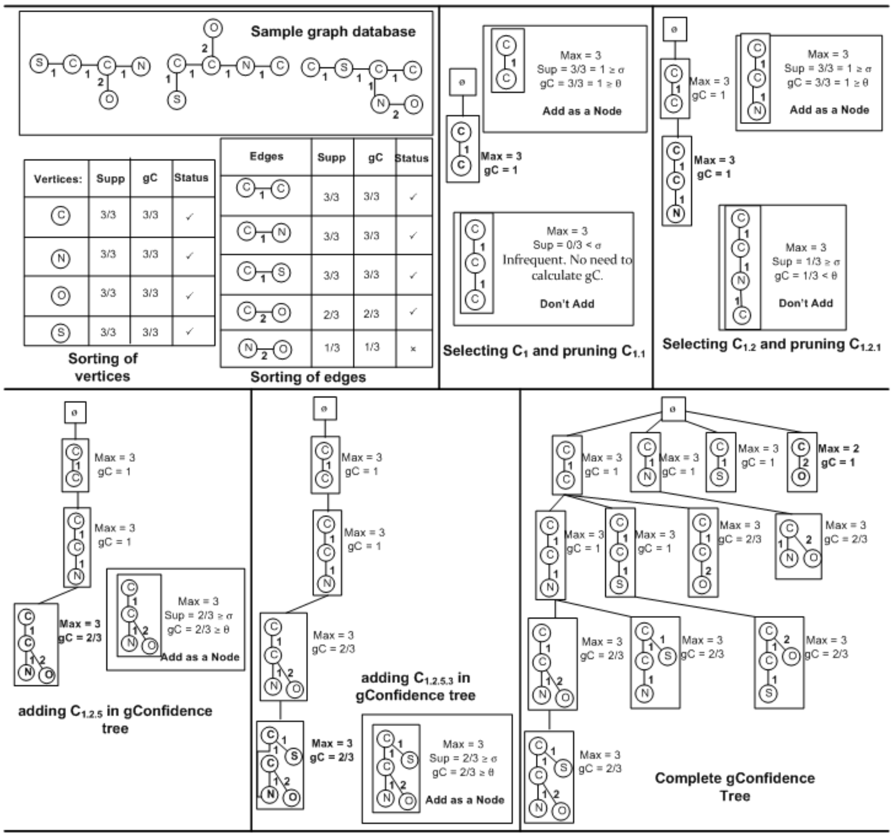
Sample graph database
Sorting of vertices
Sorting of edges
Selecting C1 and pruning C1.1
Selecting C1.2 and pruning C1.2.1
adding C1.2.5 in gConfidence tree
adding C1.2.5.3 in gConfidence tree
Complete gConfidence Tree
Infrequent. No need to calculate gC.
Add as a Node
Don't Add

Fig. 3. gConfidence mining Illustration

For illustrating the working procedure of our CGM algorithm, we can consider the graph database for the chemical dataset shown in Figure 3. We have assumed the support threshold, $\sigma = \frac{1}{3}$ and correlation threshold, $\theta = \frac{2}{3}$. According to our algorithm we have calculated *support* and *gConfidence* of each edge and vertex and then selected the frequent and correlated vertices and edges. Now, we have to construct single edge graph for each frequent-correlated edge and the edge set is also used to construct *GLOBAL EDGE MATRIX*. This matrix is sorted based on *support* count and DFS code and can be used in a chronological order for constructing potential children (smallest DFS code oriented child first) and also helps in counting maximum elementary edge support count of a graph.

We have created a Null-rooted *gConfidence* tree and then started mining for the first 1-edge graph C_1. Since it satisfies both the threshold values, we have added it in the search space and start mining the correlation of its potential children recursively. Its first child $C_{1.1}$ is not frequent, hence we have pruned it. The second child $C_{1.2}$ is frequent and correlated, hence added to the search space. However, recursive calculation of its children shows that $C_{1.2.1}$ and $C_{1.2.4}$ are frequent but not correlated and $C_{1.2.2}$ and $C_{1.2.3}$ are not frequent.

As a consequence, the fifth child of $C_{1.2}$ that is $C_{1.2.5}$ is found correlated and added to *gConfidence* tree. The eighth child of $C_{1.2.5}$ is found correlated and added in the search space but first seven children of $C_{1.2.5}$ are not frequent hence will not be correlated and been pruned from the search space. Therefore, we have to further search for the correlation of all possible children of $C_{1.2.5.8}$. Since no children of $C_{1.2.5.8}$ found frequent-correlated, we can backtrack to $C_{1.2.5}$. But already we have checked all possible children of $C_{1.2.5}$ hence we can trace back again to $C_{1.2}$ for checking correlation of its remaining children.

In this way we have calculated the correlation for the graph database of Figure 3 and found a complete *gConfidence* tree, where nodes of the tree contain correlated graphs along with the amount of correlation. All the above discussed steps are illustrated in Figure 3.

3 Experimental Results

We have performed a comprehensive performance study in our experiments on both synthetic and real world datasets. We have used our own synthetic graph generator to construct synthetic dataset. The real life datasets we have tested are cancer dataset, found from [8], namely MOLT-4 and NCI-H23. On an average, the real datasets contain about 40K graphs with average 25 nodes and 30 edges along with an average of 17 distinct labels for vertices and edges. The synthetic dataset can be identified by four properties, $\mid D \mid$ representing numbers of graphs, $\mid N \mid$ indicating number of distinct labels for vertices and edges, $\mid T \mid$ and $\mid V \mid$ for average graph size wrt. edges and vertices respectively.

In both types of data (real and synthetic) CGM algorithm is proved to be sound and efficient as well as found scalable and faster enough that it can mine correlation among various graphs with any size and any level of complexity in comparison to CGS[7] and gSpan[3]. All experiments of CGM, CGS and gSpan have been performed on a 2.1 GHz Intel(R) Core(TM) Duo PC with 1GB RAM, running Windows 7 operating system, using C/C++ programming language. We have kept the support threshold fixed at 5% and varied the correlation threshold from 45% to 85% unless stated otherwise. In some cases, we have varied the size of database by adding or removing graph transactions randomly from the actual database.

In Figure 4 and Figure 5, we have shown the performance of our proposed CGM algorithm for Processing time vs. Graph Density with varying Correlation Threshold, when run on the MOLT-4 graph database and a synthetic database characterized by $D200kN30T80V50$ respectively. Since we are assessing the performance of our algorithm against graph density, we have varied the size of the graph database. It can be noticed that maximum 150 seconds were needed and minimum required time was 25 seconds in mining real life database, where most of the time processing completes within 1000 seconds for synthetic dataset with any confidence threshold within range.

Figure 6 and Figure 7 contain the performance analysis of our proposed algorithm for Scalability wrt Time with varying Data Size, when run on the NCI-H23

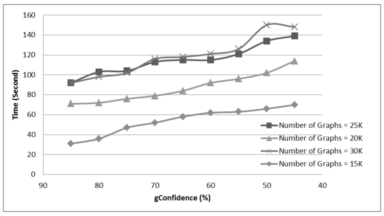

Fig. 4. Processing time wrt Graph Density on (MOLT-4)

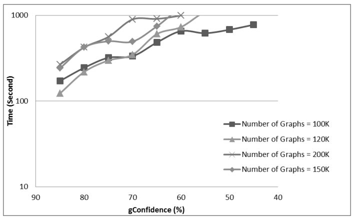

Fig. 5. Processing time wrt Graph Density on (D200kN30T80v50)

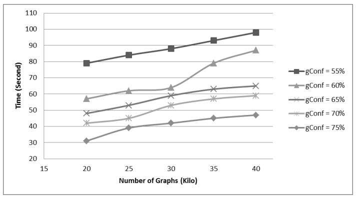

Fig. 6. Scalability wrt processing time on (NCI-H23)

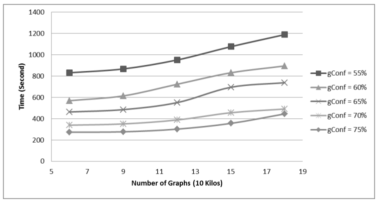

Fig. 7. Scalability of CGM wrt processing time for Synthetic Data(D200kN20T40v30)

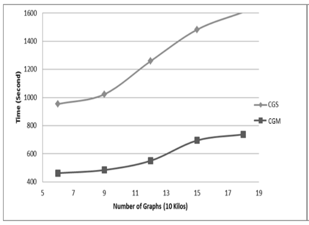

Fig. 8. CGM vs CGS on (D200kN20T40v30)

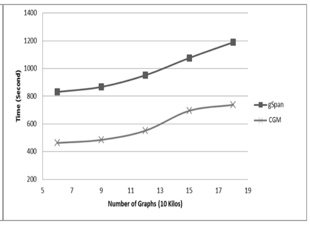

Fig. 9. CGM vs gSpan on (D200kN20T40v30)

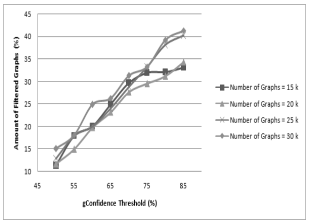

Fig. 10. Performance in Filtering(%) on (MOLT-4)

graph database and a synthtic database characterized by $D200kN20T40v30$ respectively. We have found that 50 to 100 seconds are required in mining NCI-H23 database and 300 to 1200 seconds are required for the synthetic dataset with any confidence threshold within range.

To compare the performance of CGS with CGM, once again we have used the denser synthetic dataset used in the scalability assessment earlier. However, the support threshold is considered 5% and confidence threshold 50%. The comparison is shown in the Figure 8, which illustrates a significant performance gain. We have also provided the performance of our proposed algorithm in filtering less significant graphs. By comparing it with the well known frequent subgraph mining algorithm gSpan, using denser synthetic dataset, used in the scalability assessment, we found our algorithm efficient enough in comparison with existing algorithms. Here, we have considered the support threshold 5% and confidence threshold 50%. The comparison can be found in Figure 9. Figure 10 contains the percentage of graphs, those are un-correlated, filtered by CGM with respect to the graphs selected by gSpan. Figure 10, shows that the filtering percentage of CGM can be from 10% up to 40%, on a real life data set MOLT-4, for various $gConfidence$ threshold.

4 Conclusions

Mining frequent patterns or sub-patterns with larger support threshold could miss some interesting patterns. At the same time if the threshold considered is small enough to capture such rare but interesting items could generate lots of spurious patterns. Therefore, association and correlation analysis is used in association of frequent pattern mining for mining frequent-interesting patterns from a collection of itemsets, but correlation searching is a challenging task. In a graph database, correlation searching is more challenging due to the fact that searching frequent subgraphs faces the graph isomorphism problem. Therefore, a new measure $gConfidence$ and an algorithm CGM are proposed for correlation mining in graph databases which can capture more interesting inherent correlation among graphs. $gConfidence$ has downward closure property, which helps in pruning descendants of non-correlated candidates. Proposed method constructs a tree-like search space named $gConfidence$ tree to efficiently mine the correlation. We have performed extensive performance analysis of CGM and found it efficient enough which outperforms existing works in correlation search based on speed. The proposed algorithm can be applied in both traditional and non-traditional domains such as bio-informatics, computer vision, various networks, machine learning, chemical domain and various other real life domains.

References

1. Inokuchi, A., Washio, T., Motoda, H.: An apriori-based algorithm for mining frequent substructures from graph data. In: Zighed, D.A., Komorowski, J., Żytkow, J.M. (eds.) PKDD 2000. LNCS (LNAI), vol. 1910, pp. 13–23. Springer, Heidelberg (2000)
2. Kuramochi, M., Karypis, G.: Frequent subgraph discovery. In: ICDM, pp. 313–320. IEEE Computer Society (2001)
3. Yan, X., Han, J.: gSpan: Graph-based substructure pattern mining. In: ICDM, pp. 721–724. IEEE Computer Society (2002)
4. Li, J., Liu, Y., Gao, H.: Efficient algorithms for summarizing graph patterns. IEEE Trans. Knowl. Data Eng. 23(9), 1388–1405 (2011)
5. Tan, P.N., Kumar, V., Srivastava, J.: Selecting the right interestingness measure for association patterns. In: KDD, pp. 32–41. ACM (2002)
6. Yan, X., Zhu, F., Yu, P.S., Han, J.: Feature-based similarity search in graph structures. ACM Trans. Database Syst. 31(4), 1418–1453 (2006)
7. Ke, Y., Cheng, J., Ng, W.: Efficient correlation search from graph databases. IEEE Trans. Knowl. Data Eng. 20(12), 1601–1615 (2008)
8. Pubchem web site for information on biological activities of small molecules (2011), `http://pubchem.ncbi.nlm.nih.gov`