Mining Closed Relational Graphs with Connectivity Constraints*

Xifeng Yan† X. Jasmine Zhou‡ Jiawei Han†

†Department of Computer Science, University of Illinois at Urbana-Champaign
{xyan, hanj}@cs.uiuc.edu
‡Molecular and Computational Biology, University of Southern California
xjzhou@usc.edu

1 Introduction

Relational graphs are widely used in modeling large scale networks such as biological networks and social networks. In a *relational graph*, each node represents a distinct entity while each edge represents a relationship between entities. Various algorithms were developed to discover interesting patterns from a *single* relational graph [3]. However, little attention has been paid to the patterns that are hidden in *multiple* relational graphs. One interesting pattern in relational graphs is *frequent highly connected subgraph* which can identify recurrent groups and clusters. In social networks, this kind of pattern corresponds to communities where people are strongly associated. For example, if several researchers co-author some papers, attend the same conferences, and refer their works from each other, it strongly indicates that they are studying the same research theme.

Figure 1 depicts this pattern discovery problem: *Given a set of massive relational graphs, how to mine frequent highly connected subgraphs from it?*

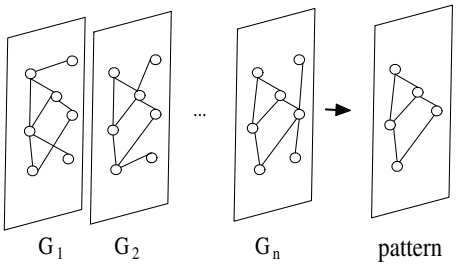

Figure 1. Mining Massive Relational Graphs

This new problem setting has three major characteristics different from the previous frequent graph mining problem [2]. First, in relational graphs each node represents a distinct object. No two nodes share the same label. In biological networks, nodes often represent unique objects like genes and enzymes. Second, relational graphs may be very large. For example, social networks often have thousands of nodes and millions of edges. Third, the interesting patterns should not only be frequent but also satisfy some connectivity constraints. Previous studies usually interpret a frequent graph as an object and ignore its internal structure such as connectivity.

2 Formulation

A *relational graph set* consists of multiple undirected simple graphs, $\{G_i = (V, E_i)\}$, $i = 1, \ldots, n$, $E_i \subseteq V \times V$, where a common vertex set V is shared among the graphs in the set. Frequent relational (sub)graphs are the subgraphs that occur frequently in a relational graph dataset. Since frequent graph mining usually generates too many patterns, it is more appealing to mine closed frequent graphs only [4]. A frequent graph g is *closed* if and only if there does not exist a supergraph with the same support. Assume graphs g and g' are both frequent subgraphs and appear in the same set of relational graphs. If g' is contained by g, then it is unnecessary to present graph g' to users since it does not provide new information.

The second concern is the connectivity of a graph pattern. An *edge cut* of a relational graph G is a set of edges E_c such that $E(G) - E_c$ is disconnected. A *minimum cut* is the smallest set in all edge cuts, whose size is written $\kappa(G)$. As suggested by many graph theoretic approaches for data clustering [3, 1], the *minimum cut measure*, also called *edge connectivity*, is good at clustering nodes based on their connectivity. In this study, we investigate the issues of *mining all closed frequent graphs with edge connectivity (minimum cut size) at least* K, where K is a positive integer. Since previous studies only provide solutions to clustering

*The work was supported in part by the U.S. National Science Foundation under grants IIS-02-09199/IIS-03-08215, an IBM Faculty Award, and a USC Faculty Start Up Fund.

objects in one graph, instead of a set of graphs, we have to solve two issues emerging from this new graph mining problem: (1) how to mine closed frequent graphs efficiently in large relational graphs, and (2) how to handle the connectivity constraint.

We briefly introduce our solution to the above two issues. Different from general labeled graphs, relational graphs have unique label for each node. Because of this special property, we can treat relational graphs as sets of edges and use the *closed frequent itemset* mining technique instead of general graph mining algorithms. Here, each distinct edge is an item. The mining algorithm should assure each discovered graph pattern connected.

There is no downward closure property for edge connectivity. That is, the high connectivity of a graph does not imply the high connectivity of its supergraph, and vice versa. Therefore, we have to enumerate all subgraphs of each closed frequent graph and check their connectivity. Since billions of frequent graphs may exist in one dataset, it is impossible to finish this brute-force computation within limited time. Thus, we develop two graph theoretic approaches, CLOSECUT (a pattern-growth approach) and SPLAT (a pattern-reduction approach), to efficiently discover closed highly connected graphs. CLOSECUT extends a candidate graph by inserting new edges until the candidate graph is not frequent any more, while SPLAT works in the reverse direction. Furthermore, we apply graph condensation and decomposition techniques in the design of CLOSECUT and SPLAT to improve the performance. Both of them can reduce the size of candidate graphs in terms of nodes and edges.

3 Experimental Results

We conducted a comprehensive performance study on both synthetic and real datasets. The synthetic data is controlled by a set of parameters that allow us to test the performance under different conditions. The real dataset is extracted from microarray data.

Figure 2 shows the runtime of CLOSECUT and SPLAT for a synthetic dataset. This dataset contains a small number of relational graphs, each with a large number of nodes. Each graph in this dataset contains multiple highly connected subgraphs, and these subgraphs overlap with each other. As shown in the figure, CLOSECUT and SPLAT have the similar performance when the support is high. The high support threshold filters out lots of infrequent edges and noise edges. Thus, both algorithms complete very fast. When the support is lowered down, CLOSECUT outperforms SPLAT because SPLAT has to enumerate lots of infrequent highly connected subgraphs, which will eventually be discarded. However, when the support is very low, the situation is reversed. SPLAT can use the connectivity constraint to remove many frequent, but low minimum cut edges.

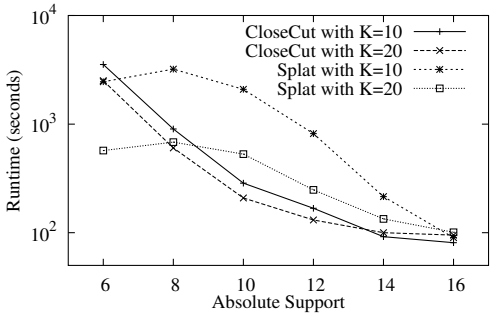

Figure 2. Runtime vs. Support

We also applied CLOSECUT and SPLAT in biological datasets and successfully identified interesting patterns from multiple biological networks. In the full version of this paper, we will formulate CLOSECUT and SPLAT, and report the patterns they discovered systematically.

4 Conclusions

In this paper, we introduced a new graph mining problem: *finding closed frequent graphs with connectivity constraints in relational graphs*. We adopted the concept of edge connectivity and applied graph theoretic results, graph condensation and decomposition, in our algorithm design. Through our study, a new research area in frequent graph mining is exposed, where the previous algorithms on single graph mining should be re-examined for pattern discovery in multiple relational graphs.

References

[1] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[2] T. Washio and H. Motoda. State of the art of graph-based data mining. *SIGKDD Explorations*, 5:59–68, 2003.

[3] Z. Wu and R. Leahy. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 15:1101–1113, 1993.

[4] X. Yan and J. Han. CloseGraph: Mining closed frequent graph patterns. In *Proc. 2003 ACM Int. Conf. on Knowledge Discovery and Data Mining (KDD'03)*, pages 286–295, Aug. 2003.