Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 869198, 6 pages
http://dx.doi.org/10.1155/2014/869198

Review Article

Performance Evaluation of Frequent Subgraph Discovery Techniques

Saif Ur Rehman,[1] Sohail Asghar,[2] Yan Zhuang,[3] and Simon Fong[3]

[1] *UIIT, PMAS Arid Agriculture University, Rawalpindi, Pakistan*
[2] *COMSATS Islamabad, Pakistan*
[3] *Department of Computer and Information Science, University of Macau, Macau*

Correspondence should be addressed to Saif Ur Rehman; saifi.ur.rehman@gmail.com and Yan Zhuang; syz@umac.mo

Received 7 May 2014; Accepted 28 June 2014; Published 6 August 2014

Academic Editor: Erik Cuevas

Copyright © 2014 Saif Ur Rehman et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Due to rapid development of the Internet technology and new scientific advances, the number of applications that model the data as graphs increases, because graphs have highly expressive power to model a complicated structure. Graph mining is a well-explored area of research which is gaining popularity in the data mining community. A graph is a general model to represent data and has been used in many domains such as cheminformatics, web information management system, computer network, and bioinformatics, to name a few. In graph mining the frequent subgraph discovery is a challenging task. Frequent subgraph mining is concerned with discovery of those subgraphs from graph dataset which have frequent or multiple instances within the given graph dataset. In the literature a large number of frequent subgraph mining algorithms have been proposed; these included FSG, AGM, gSpan, CloseGraph, SPIN, Gaston, and Mofa. The objective of this research work is to perform quantitative comparison of the above listed techniques. The performances of these techniques have been evaluated through a number of experiments based on three different state-of-the-art graph datasets. This novel work will provide base for anyone who is working to design a new frequent subgraph discovery technique.

1. Introduction

The increasing use of large communication, financial, telecommunication, and social networks is giving birth to substantial source of problems for the graph data mining researchers [1]. These systems are different from many other traditional data mining problems in that the data records representing transactions between set of entities are not considered independent, and the intertransaction dependencies can be represented as trees, lattices, sequences, and graphs [2]. As a result, there has been an increasing interest in analyzing the properties, models, and algorithms which are applicable to graph-structured data to address such type of issues [1].

In the recent years, the data mining techniques are facing the challenge to cope with an ever growing number of complex types of objects [3, 4]. Graphs structure is considered a useful natural data structure which can be easily applied to model such type of complex objects [3, 4]. Graphs have become increasingly useful in modeling complicated structures and schema-less data including proteins structures, electrical circuits, images, Web data, and XML based documents [5]. Conceptually, any kind of data can be represented by graphs. In addition to the ubiquitous use of XML in Web information, we also witnessed the wide usage of graph databases in a number of other different domains. For example, in computer vision, these are used to represent complex relationships, such as the organization of entities in images. These relationships can be used to identify objects and scenes. Efficient retrieval of graph-based models is an essential problem in pattern recognition, as indicated by the wealth of research literature [5]. In chemical informatics and bioinformatics, scientists use graphs to represent compounds and proteins. Daylight system [6] which is a commercial product for compound registration has already been used

in chemical informatics. Benefiting from such a system, researchers are able to do screening, designing, and knowledge discovery from compound or molecular databases.

Graph mining is becoming an important topic of research recently because of numerous applications to a wide variety of data mining problems including computational biology, chemical data analysis, drug discovery, and communication networking. Traditional data mining techniques such as clustering, classification, frequent pattern mining, and indexing have now been extended to the graph scenario [7].

Recently, there arise a large number of graphs with massive size and complex structures in many new application domains, such as biological networks, social networks, and the information flow pattern on the Internet demanding the powerful data mining techniques. The major contribution areas in our study included the following.

(i) We have summarized six well-known frequent subgraph algorithms proposed so far and have reimplemented these techniques for performance evaluation purposes on the same platform using the same expertise level.

(ii) We have compared the performance of these techniques based on different setting of support threshold, required for execution of these methods.

(iii) Our work is superior to others in the literature as it has evaluated the performance of multiple frequent graph techniques based on the various dataset of varying size.

(iv) We have concluded this with some recommendations based on which some new frequent subgraph algorithms will be developed by the researchers in this very grooming field.

2. Graph Preliminaries

In this section, some basic concepts of graph theory have been defined as well as the notation used in the rest of this study.

2.1. Graph. A graph is a pair $G = (V, E)$ consisting of a set of vertices V and a set of edges $E \sqsubseteq VXV$ such that every edge $e \in E$ relates to a pair of vertices (v_1, v_2).

2.2. Graph Isomorphism. Two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are isomorphic if they are topologically identical to each other; that is, there is a mapping from G_1 to G_2 such that each edge in E_1 is mapped to a single edge in E_2 and vice versa. In the case of labeled graphs, this mapping must also preserve the labels on the vertices and edges.

2.3. Subgraph. A graph $G_2 = (V_2, E_2)$ is a subgraph of another graph $G_1 = (V_1, E_1)$ iff $V_2 \subseteq V_1$, and $E_2 \sqsubseteq E_1 \wedge (v_1, v_2) \in E_2 \rightarrow V_1 \in V_2$ and $v_2 \in V_2$ (Figure 1).

2.4. Induced Subgraph. Let $G_1 = (V_1, E_1, \alpha_1, \beta_1)$ and $G_2 = (V_2, E_2, \alpha_2, \beta_2)$ be graphs. G_2 is an induced subgraph of G_1, $(G_2 \subseteq G_1)$, if $V_2 \subseteq V_1$, $\alpha_1(v) = \alpha_2(v)$ for all $v \in V_2$, $E_2 = E_1 \cap (V_2 \times V_2)$, and $\beta_1(e) = \beta_2(e)$ for all $e \in E_2$. Given a graph $G_1 = (V_1, E_1, \alpha_1, \beta_1)$, if any subset $V_2 \in V_1$ of its vertices uniquely defines a subgraph, this subgraph is called the subgraph induced by V_2.

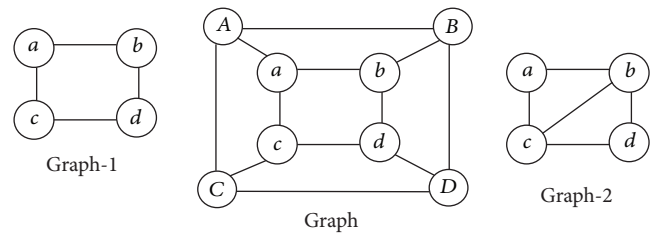


Figure 1: Graph-1 is subgraph of Graph but Graph-2 is not subgraph of Graph.

2.5. Subgraph Isomorphism. Given two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, the problem of subgraph isomorphism is to find an isomorphism between G_2 and a subgraph of G_1, that is, to determine whether or not G_2 is included in G_1.

2.6. Frequent Subgraph. A frequent subgraph is a graph whose support is not less than a minimum user-specified support threshold. Given a labeled graph dataset $G_D = \{G_1, G_2, \ldots, G_k\}$, support or frequency of a subgraph g is the percentage (or number) of graphs in G_D where g is a subgraph.

3. Frequent Subgraph Mining (FSM)

The frequent subgraph is defined as a graph which occurs frequently in the graph database; graph database is special type of database that mostly includes a single large graph or some multiple small graphs. But in the area of biological networks, database is consisting of some large graph [8]. Indeed this issue can be explained exactly as follows.

If D is the entry database, the frequent subgraph mining aims to mine graphs with more support value in comparison with predetermined threshold. The graph support G_S is denoted by sup (G_S) and is given as

$$\text{Sup}\,(G_S) = \frac{\sum_{i=1}^{n} G_i}{n}. \tag{1}$$

In (1), n is the total number of graphs in the graph dataset. An example of frequent subgraph mining is given here. Two graphs, say g_1 and g_2, from graph database have been used as input and third g_3 is used as output as a frequent subgraph (Figure 2).

In the last decade, the frequent subgraph discovery has attracted much attention in the graph mining community with various efficient algorithms developed such as FSG [9], AGM [10], gSpan [11], CloseGraph [12], SPIN [3, 4], Gaston [13], and Mofa [14]. These algorithms have worked on efficient mining of complete pattern sets. These techniques aim to identify the frequently occurring patterns from a given collection of small graphs or within one large graph. Frequent subgraphs discovery algorithms have been extensively used

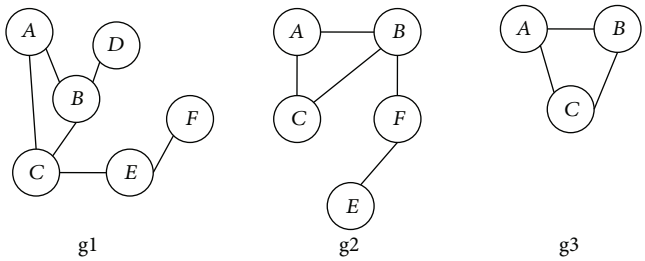


Figure 2: g1 and g2 are input database graph and g3 is output as frequent subgraph [16].

in wide spread application domains such as computational biology, which include discovering the motif structure for protein interaction and finding the residue packing patterns from protein structure and chemical carcinogenesis. These algorithms are found to be useful in security threat assessment and analysis as well [15].

In most of the literatures [3, 4], FSM techniques have been classified into two major categories: (1) apriori-based approaches and (2) pattern growth-based approach. The techniques under the umbrella of apriori-based frequent substructure mining algorithms share similar characteristics with apriori-based frequent item-set mining algorithms [3, 4]. The apriori-based algorithms have considerable overhead when two size-k frequent substructures are joined to generate size-$(k + 1)$ graph candidates. The apriori-based subgraph mining techniques included [3, 4, 9, 10, 17].

To overcome the overhead of apriori algorithms, non-apriori-based algorithms have been developed, most of which adopt the pattern-growth methodology [3, 4]. The pattern-growth mining technique extends a frequent graph by adding a new edge, in every possible position [3, 4]. These techniques are facing the problem that the same graph can be discovered many times with the edge extension. These techniques included [11–14].

Algorithms in the second category use a depth-first search to enumerate candidate frequent subgraphs [2, 18–20]. As demonstrated in these papers, depth first algorithms provide advantages over level-wise search for (1) better memory utilization and (2) efficient subgraph testing; for example, it usually permits the subgraph test to be performed incrementally at successive levels during the search [17].

gSpan: Graph-Based Substructure Pattern Mining. The gSpan is the state-of-the-art subgraph mining algorithm, with the aim of discovering the frequent substructure without candidate generation and false pruning. This algorithm was proposed by [11]. The gSpan is the first algorithm that introduced the depth-first search in FSM.

gSpan uses a canonical representation for graphs, called *DFS*-Code. A *DFS-traversal* of a graph defines an order in which the edges are visited. The concatenation of edge representations in that order is the graph's *DFS-Code*. Refinement generation is restricted by *gSpan* in two ways. First, fragments can only be extended at nodes that lie on the rightmost path of the *DFS-tree*. Secondly, fragment generation is guided by occurrence in the appearance lists. Since these two pruning rules cannot fully prevent isomorphic fragment generation, gSpan computes the canonical (lexicographically smallest) *DFS-Code* for each refinement by means of a series of permutations. Refinements with nonminimal *DFS-Code* can be pruned. Since, instead of embeddings, *gSpan* only stores appearance lists for each fragment, explicit subgraph isomorphism testing must be done on all graphs in these appearance lists [11] Algorithm 1.

4. Subgraph Discovery Algorithms Based on gSpan

In this section, a number of frequent subgraph discovery algorithms have been briefly described.

4.1. FFSM (Fast Frequent Subgraph Mining). FFSM represents graphs as triangle matrices (node labels on the diagonal, edge labels elsewhere). The matrix-code is the concatenation of all its entries, left to right and line by line. Based on lexicographic ordering, isomorphic graphs have the same canonical code (CAM—Canonical Adjacency Matrix). When FFSM joins two matrices of fragments to generate refinements, only at most two new structures result. FFSM also needs a restricted extension operation: a new edge-node pair may only be added to the last node of a CAM. After refinement generation, FFSM permutes matrix lines to check whether a generated matrix is in canonical form. If not, it can be pruned. FFSM stores embeddings to avoid explicit subgraph isomorphism testing. However, FFSM only stores the matching nodes, and edges are ignored. This helps in speeding up the join and extension operations since the embedding lists of new fragments can be calculated by set operations on the nodes. As an important shortcoming of the method, FFSM requires checking the frequency of an additional set of subgraphs that are not canonical [17].

4.2. Gaston (GrAph/Sequence/Tree extractiON). The Gaston keeps all the embeddings in order to generate only refinements which actually appear and to achieve fast isomorphism testing. The main insight is that there are efficient ways to enumerate paths and (noncyclic) trees. By considering fragments that are paths or trees first, and by only proceeding to general graphs with cycles at the end, a large fraction of the work can be done efficiently. Only in that last phase, Gaston faces the *NP-completeness* of the subgraph isomorphism problem. Gaston defines a global order on cycle-closing edges and only generates those cycles that are "*larger*" than the last one. In Gaston, the detection of duplicate is achieved in two steps: in step one hashing is done to presort and in second step a graph isomorphism test is done in order to find the final duplicate [13].

4.3. CloseGraph (Mining Closed Frequent Graph Patterns). CloseGraph focuses on mining of closed subpattern instead of mining all the subgraph. The authors have defined the close graph as "*A graph g is closed graph in a database if there exists no proper supergraph of g that has the same support*

GraphSet Projection (D, S).
(1) sort the labels in D. by their frequency;
(2) remove infrequent vertices and edges;
(3) re-label the remaining vertices and edges;
(4) $S1 \leftarrow$ all frequent 1-edge graphs in D;
(5) sort $S1$ in DFS lexicographic order;
(6) $S1 \leftarrow S$
(7) for each edge e in $S1$ do
(8) initialize s with e, set s. D by graphs which contains e;
(9) Subgraph Mining(D, S, s);
(10) $D \leftarrow D - e$
(11) if $|D| <$ minsup;
(12) break;

Sub_procedure 1 Subgraph Mining(D, S, s)
(1) if s # min(s)
(2) return;
(3) $S \leftarrow SU\{s\}$;
(4) enumerate s in each graph in D. and count its children;
(5) for each c, cN is s' child do
(6) if support(c) grtequalminsup
(7) $s \leftarrow c$:
(8) Subgraph Mining$(.D, S, s)$;

Algorithm 1: gSpan [11] algorithm.

as g." The basic theme of the CloseGraph is based on the state-of-the-art gSpan and formulates the early termination condition to make gSpan "return" as soon as possible in case of close graph mining. The two main concepts of CloseGraph are (I) right most extension and DFS (depth-first-search) lexicographic order for frequent graph generation and (II) equivalent occurrence and early termination for nonclosed graphs pruning. The details can be found in [15]. From the experimental results, authors have shown that their technique not only dramatically reduces the generation of unnecessary subgraph generation but also increases the efficiency of mining the graph databases, especially in the presence of large graph sets.

4.4. SPIN (Mining Maximal Frequent Subgraphs from Graph Databases). In large graph databases, the total number of frequent subgraphs can become too large to allow a full enumeration using reasonable computational resources. In this paper, authors have proposed a new algorithm that mines only maximal frequent subgraphs; that is, subgraphs that are not a part of any other frequent subgraphs. This may exponentially decrease the size of the output set in the best case. SPIN is the novel FSM technique to mine only maximal frequent subgraphs of large graph databases. It has integrated several optimization techniques to speed up the mining process [18, 19].

4.5. FSP: Frequent Substructure Pattern Mining. Han et al. have proposed some enhancement in the gSpan subgraph discovery algorithm and termed their technique as FSP. This new enhancement in gSpan has achieved the following success in improving the gSpan subgraph discovery: (1) improving the canonical graph representation used in gSpan and defining relationship between sub-DFS trees whose root shares the same parent node in the candidate subgraphs; (2) discovering the structural similarities in the DFS Search Space and presenting the two one-to-one reflection between children subgraph and latter sibling subgraphs; (3) applying the two techniques to subgraph mining problem to reduce the number of graph and subgraph isomorphism tests effectively; and (4) implementing the FSP algorithm and other current related techniques and comparing the results.

5. Results and Discussion

In the following sections we have evaluated the performance of the FSM techniques discussed in the study. Experiments were carried out on the datasets, whose statistical information is given in Table 1.

All our experiments have been performed on a 32-bit Linux system with 4 GB memory and 3.0 GHz Intel processor. Executable of gSpan, Gaston has been obtained from the respective authors and remaining techniques have been reimplemented. The performance of the discussed frequent subgraph discovery algorithms has been evaluated on the three datasets except gSpan whose performance is evaluated with respect to other techniques on one dataset, that is, chemical compound dataset, as for the dataset where the number of graph increases 400, the gSpan took much time. For this reason we have compared all other techniques on three datasets except gSpan.

The results obtained are shown in Figures 3, 4, and 5 at various threshold values, that is, 10, 20, 30, 40, and 50, which clearly indicates that execution time of these FSM techniques

Table 1: Dataset statistics.

Dataset	Description of the dataset used in the experiments
Chemical compound	340 chemical compounds, 24 different atoms, 66 atom types, and 4 types of bonds [5]
AIDS antiviral screen compound	The dataset contains 43,095 chemical compounds [12]
DTP human tumor cell line screen (CANSO3SD)	It consists of 42,247 molecules. Each molecule corresponds to a graph, where atoms are represented using nodes and the bonds between them are represented by edges

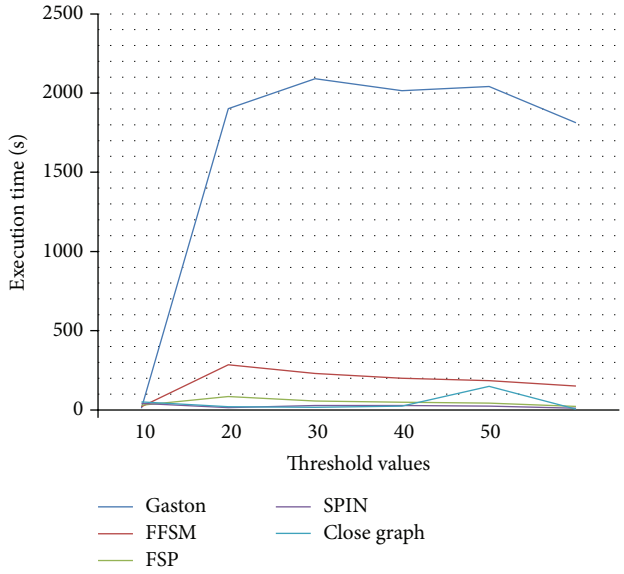


Figure 3: Comparison of various FSM techniques on chemical dataset (time is given Log_2 for clarification).

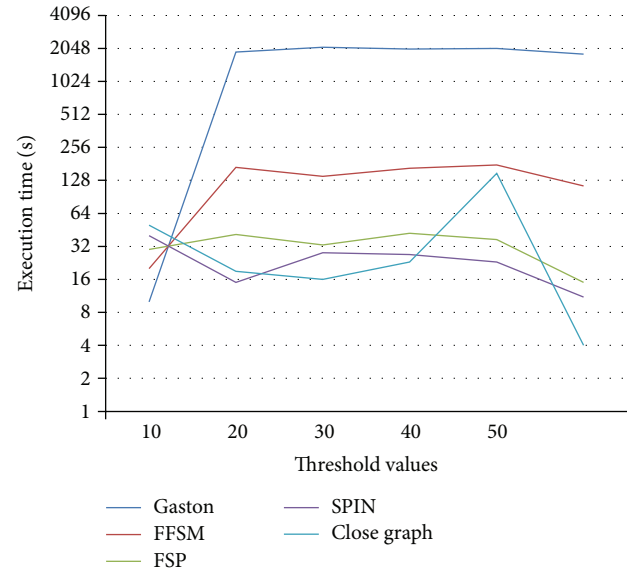


Figure 4: Comparison of various FSM techniques on AIDS antiviral screen compound.

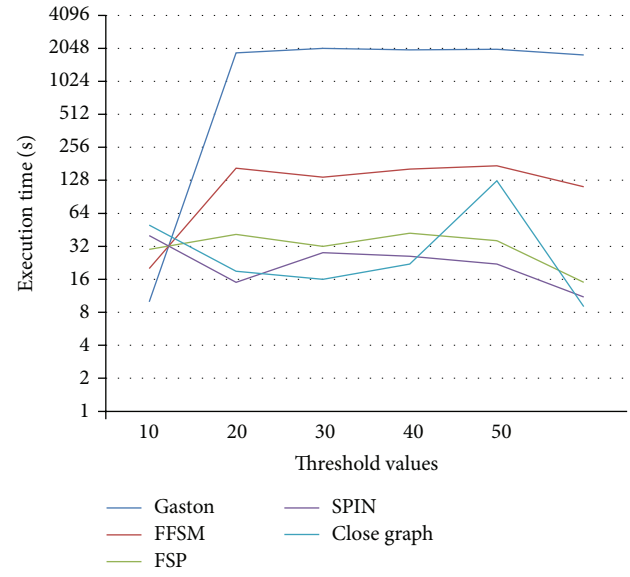


Figure 5: Comparison of various FSM techniques on DTP human tumor cell line screen (CANSO3SD) (time is given Log_2 for clarification).

decreases as we increase the support threshold. We have observed that time of execution is inversely proportional to the support threshold, that is, as the support threshold is increased the execution time of the technique is reduced. It is very clear that when the support threshold is fixed to 50 then all of these techniques give the best output in terms of execution time. Moreover, we can conclude that FSP gives best performance as compared to the other frequent subgraph discovery as indicated in the result shown in Figure 3. The performance of CloseGraph is also good but it can only be used for the closed subgraphs datasets.

6. Conclusion and Future Directions

Graph mining is a well-explored area of research in the data mining community in which the frequent subgraph discovery is one of the most challenging problems. In literature, a large number of frequent subgraph discovery algorithms have been proposed. This work has provided a comprehensive description of some very prominent and efficient subgraph discovery algorithms along with their basic implementation algorithms. In this study, we have provided a comprehensive comparison of various FSM techniques and reimplemented these techniques. Furthermore, we have provided the performance evaluation of various famous subgraph discovery algorithms which shows their relative efficiency; this will further help researchers in the graph mining community to choose the technique that best suits their problems. The supremacy of this research study is that it has brought most of the frequent subgraph algorithms under one umbrella and has evaluated their performance on multiple datasets

of different sizes. This work can be used as a base for the introduction of the new FSM algorithm.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The authors would like to thank Mr. Luke Huan in the University of Kansas for sharing and giving support in FFSM source code. They also thank Mr. Xifeng Yan and Professor Jiawei Han in University of Illinois at Urbana Champaign for providing the gSpan executable. The authors are also thankful for the financial support from the research grant of Grant no. MYRG152(Y3-L2)-FST11-ZY, offered by the University of Macau, RDAO.

References

[1] R. Vijayalakshmi, R. Nadarajan, J. F. Roddick, M. Thilaga, and P. Nirmala, "FP-graphminer - A fast frequent pattern mining algorithm for network graphs," *Journal of Graph Algorithms and Applications*, vol. 15, no. 6, pp. 753–776, 2011.

[2] S. Han, K. N. Wee, and Y. Yu, "FSP: frequent substructure pattern mining," in *Proceedings of the 9th International Conference on Information and Communication Security (ICICS '07)*, IEEE, Zhengzhou, China, December 2007.

[3] K. Lakshmi and T. Meyyappan, "A comparative study of frequent subgraph mining algorithms," *International Journal of Information Technology Convergence and Service*, vol. 2, no. 2, p. 23, 2012.

[4] K. Lakshmi and T. Meyyappan, "Frequent subgraph mining algorithms—a survey and framework for classification," in *Proceedings of the Conference on Innovations in Theoretical Computer Science (ITCS '12)*, pp. 189–202, 2012.

[5] X. Yan, P. S. Yu, and J. Han, "Graph indexing: a frequent structure-based approach," in *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD '04)*, pp. 335–346, Paris, France, June 2004.

[6] C. A. James, D. Weininger, and J. Delany, *Daylight Theory Manual Daylight Version 4.82*, Daylight Chemical Information Systems, 2003.

[7] C. C. Aggarwal and H. Wang, *Managing and Mining Graph Data*, Springer, 1st edition, 2010.

[8] X. Wu, L. Jain, T. L. Wang, M. J. Zaki, H. T. T. Toivonen, and D. Shasha, *Data Mining in Bioinformatics*, Springer, London, UK, 2005.

[9] M. Kuramochi and G. Karypis, "Frequent subgraph discovery," in *Proceedings of the 1st IEEE International Conference on Data Mining (ICDM '01)*, pp. 313–320, Piscataway, NJ, USA, December 2001.

[10] A. Inokuchi, T. Washio, and H. Motoda, "An apriori-based algorithm for mining frequent substructures from graph data," in *Proceeding of the Practice of Knowledge Discovery in Databases Conference (PKDD '00)*, pp. 13–23, 2000.

[11] X. Yan and J. Han, "gSpan: graph-based substructure pattern mining," in *Proceedings of the 2nd IEEE International Conference on Data Mining (ICDM '02)*, pp. 721–724, Maebashi City, Japan, December 2002.

[12] X. Yan and J. Han, "CloseGraph: mining closed frequent graph patterns," in *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '03)*, pp. 286–295, ACM Press, Washington, DC, USA, August 2003.

[13] S. Nijssen and J. N. Kok, "A quickstart in frequent structure mining can make a difference," in *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '04)*, pp. 647–652, August 2004.

[14] C. Borgelt and M. R. Berthold, "Mining molecular fragments: finding relevant substructures of molecules," in *Proceedings of the 2nd IEEE International Conference on Data Mining (ICDM '02)*, pp. 51–58, Maebashi, Japan, December 2002.

[15] V. Krishna, N. N. R. Ranga Suri, and G. Athithan, "A comparative survey of algorithms for frequent subgraph discovery," *Current Science*, vol. 100, no. 2, pp. 190–198, 2011.

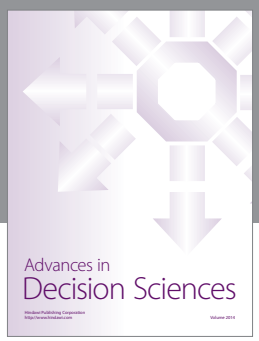
[16] J. K. Bhut and M. Vithalani, "Review on frequent subgraph pattern mining algorithms," *International Journal of Engineering Research & Technology*, vol. 2, no. 10, 2013.

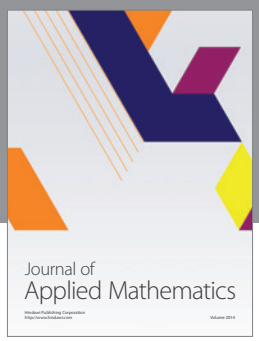
[17] J. Huan, W. Wang, and J. Prins, "Efficient mining of frequent subgraphs in the presence of isomorphism," in *Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM '03)*, pp. 549–552, Piscataway, NJ, USA, November 2003.

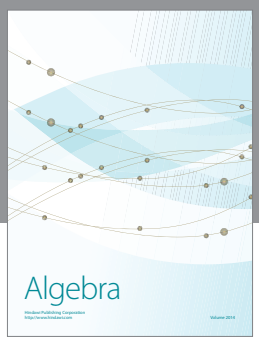
[18] J. Huan, W. Wang, D. Bandyopadhyay, J. Snoeyink, J. Prins, and A. Tropsha, "Mining protein family specific residue packing patterns from protein structure graphs," in *Proceedings of the 8th Annual International Conference on Research in Computational Molecular Biology (RECOMB '04)*, pp. 308–315, San Diego, Calif, USA, March 2004.

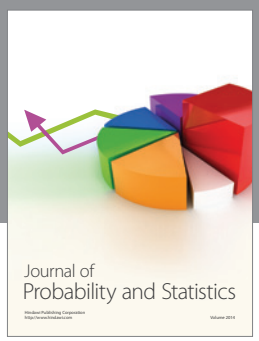
[19] J. Huan, W. Wang, J. Prins, and J. Yang, "SPIN: mining maximal frequent subgraphs from graph databases," in *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '04)*, pp. 581–586, Seattle, Wash, USA, August 2004.

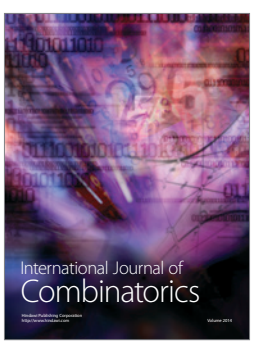
[20] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann, 1st edition, 2000.

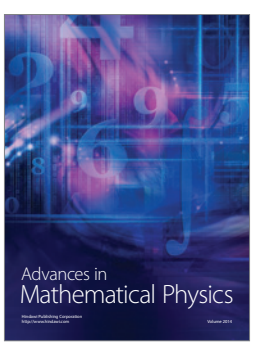


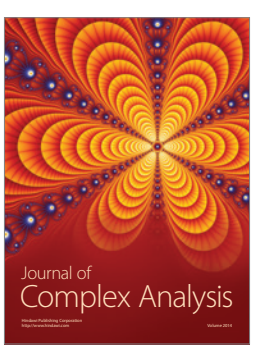


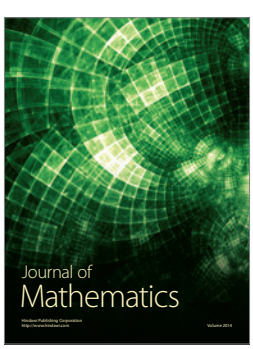


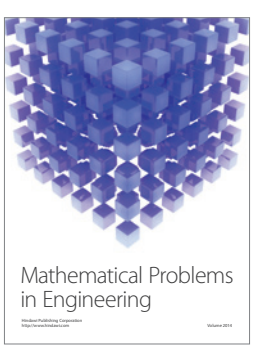


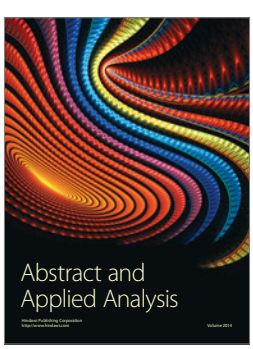


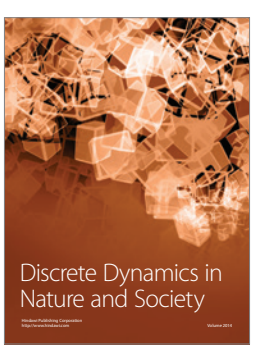


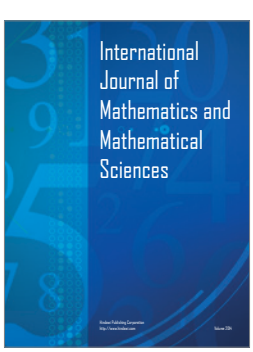












Hindawi

Submit your manuscripts at
http://www.hindawi.com