International Journal of Engineering Inventions
ISSN: 2278-7461, www.ijeijournal.com
Volume 1, Issue 5 (September2012) PP: 60-63

A Survey on Algorithms of Mining Frequent Subgraphs

Maryam Gholami [1], Afshin Salajegheh[2]
[1,2]*Department of COMPUTER Eng., South Tehran Branch, Islamic Azad University, Tehran, Iran*

Abstract—Graphs are currently becoming more important in modeling and demonstrating information. In the recent years, graph mining is becoming an interesting field for various processes such as chemical compounds, protein structures, social networks and computer networks. One of the most important concepts in graph mining is to find frequent subgraphs. The major advantage of utilizing subgraphs is speeding up the search for similarities, finding graph specifications and graph classifications. In this article we classify the main algorithms in the graph mining field. Some fundamental algorithms are reviewed and categorized. Some issues for any algorithm are graph representation, search strategy, nature of input and completeness of output that are discussed in this article.

Keywords—Frequent subgraph, Graph mining, Graph mining algorithms

I. INTRODUCTION

Extraction of useful and new information from data is defined as data mining. The aim of data mining is to discover beneficial algorithms hidden in a data set and using them for explicit knowledge. With the increasing amount and complexity of today's data, there is an urgent need to accelerate data mining for large databases. Furthermore, most of the data is either structural in nature, or is composed of parts and relations between those parts. Hence, there exists a need for developing scalable tools to analyze and discover concepts in a structural database. Graphs can represent a wide spectrum of data. However, they are typically used when relationships are crucial to the domain. A Graph, as a general data structure, can be used to model many complicated relations among the data. Labels for vertices and edges can represent different attributes of entities and relations among them. For instance, in chemical compounds, the labels for vertices and edges can be different types of atoms and bounds, respectively. In electronic transactions, labels for vertices represent account owners while occurrence of payments is indicated by edges between these owners. Since graph can model more complicated objects, it has been widely used in chemical information, text retrieval, computer vision, and video indexing. Graph-based data mining or graph mining is defined as the extraction of novel and useful knowledge from a graph representation of data. In recent years, graph mining has become a popular area of research due to its numerous applications in a wide variety of practical fields such as sociology, software bug localization, and computer networking. Graph mining is an important tool to transform the graphical data into graphical information. One of the most important concepts in graph mining is to find frequent subgraphs in it, i.e. to find graphs that are repeated in the main graph (Fig. 1). Frequent subgraph mining delivers effective structured information such as common protein structures and shared patterns in object recognition. It can also be utilized in fraud detection to catch similar fraud transaction patterns from millions of electronic payments.

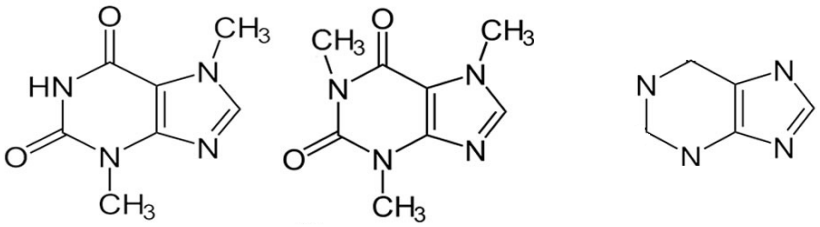

Fig. 1. From left to right: theobromine, caffeine and frequent subgraph.

II. APPROACHES OF GRAPH MINING

Graph mining algorithms can be divided in three main categories of Greedy search, Inductive logic programming and Graph theory as shown in Fig. 2. These three categories are discussed briefly in the following sections.

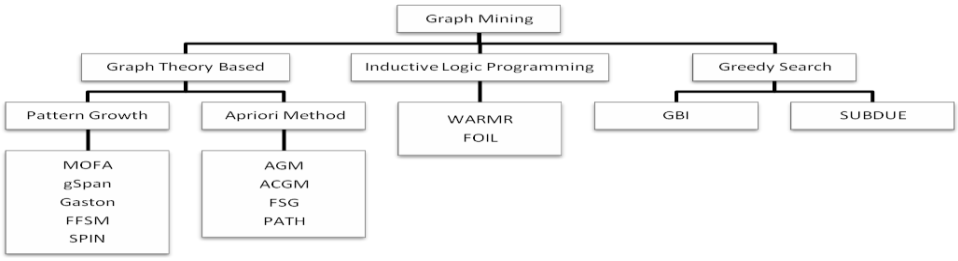

Fig. 2. Categorization of Graph mining algorithms

2-1- Greedy search approaches

In these approaches at each stage we make a decision that appears to be best at the time. A decision made at one stage is not altered in later stages and thus, each decision should assure feasibility. The two pioneering works in this field are SUBDUE [1] proposed by Cook et al and GBI[8] proposed by Yoshida et al. these two algorithms can use powerful measures such as information entropy, gini-index and description length to figure out important graph structures[4]. SUBDUE is an algorithm to find a subgraph which can maximally compress graph G based on MDL [1]. MDL is a theory in which the best hypothesis for a given set of data is the one that leads to the best compression of the data. Usually a code is fixed to compress the data, most generally with a computer language.

The motivation for this process is to find substructures capable of compressing the data, which is done by abstracting the substructure and identify conceptually interesting substructures that enhance the interpretation of the data. In this method a threshold of similarity is defined which is the cost of changes in one graph to isomorphs it with another one. There could be some little differences between the samples. SUBDUE's search is guided by the MDL principle given in equation (1) where DL(S) is the description length of the substructure being evaluated, DL (G|S) is the description length of the graph as compressed by the substructure, and DL(G) is the description length of the original graph. The best substructure is the one that minimizes this compression ratio:

$$Compression = \frac{DL(S) + DL(G|S)}{DL(G)} \qquad (1)$$

Where DL(G) is a measure of the minimum number of bites of information needed to represent G.

2-2- Inductive Logic Programming approaches

In the Inductive Logic Programming (ILP) [9] System represents predicates combining examples and background knowledge. The core of ILP is to use of logic for representation and the search for syntactically legal hypothesis constructed from predicates provided by the background knowledge.

2-3- Graph theory approaches

These approaches mine a complete set of subgraph mainly using a support or a frequent measure. Frequent subgraph is a graph with a frequency more than that of the minimum support. Mining frequent substructures usually consists of two steps. First, the frequent substructure candidates are generated. Second, the frequency of each candidate frequency is checked. Graph based approaches are mainly divided into Apriori based approaches and pattern growth approaches.

2-3-1- Apriori based algorithms

In these approaches before mining any of the subgraphs of size k+1, mining of subgraphs with size k needs to be completed [3]; where the size of the graph is defined by the number of its vertices. In this method, since the candidates are generated in level-wise manner, the BFS search strategy must be used. To generate the biggest candidate for frequent subgraphs, frequent subgraphs of size k are merged together to generate a graph of size k+1. Major algorithms with this approach are AGM[1] proposed by Inokuchi et al in 1998 [3], FSG[2] proposed by Kuramochi et al in 2001 [7] and PATH proposed by Vanetik in 2002 [10]. In the AGM algorithm, the graph is presented by an adjacency matrix [4]. First, this matrix is sorted by the lexicographic order of the labels of the vertices to find a unique form. Then, the code of the adjacency matrix is defined as shown in Fig. 3 to reduce the memory consumption.

$$X_k = \begin{pmatrix} 0 & x_{1,2} & x_{1,3} & \cdots & x_{1,k} \\ x_{2,1} & 0 & x_{2,3} & \cdots & x_{2,k} \\ x_{3,1} & x_{3,2} & 0 & \cdots & x_{3,k} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{k,1} & x_{k,2} & x_{k,3} & \cdots & 0 \end{pmatrix},$$

$$code(X_k) = x_{1,2}x_{1,3}x_{2,3}x_{1,4}\cdots x_{k-2,k}x_{k-1,k}.$$

Fig. 3. Code of the adjacency matrix

Two graphs of size k can join only if they have k-1 equal elements and the first matrix has a code less or equal than the second one. After they are joined, a graph with a normal form is generated. As the normal form representation in general is not unique, canonical form is defined as the one which has the minimum code. The canonical form is used to count the frequency of frequent subgraph.

Apriori-like algorithms suffer two considerable overheads: (1) the generation of size k+1 subgraph candidates from size k is costly and complicated, (2) Subgraph isomorphism test is an NP-Complete problem, which means that the problem could not be solved in a polynomial time. Due to these limitations, non-Apriori algorithms are developed.

2-3-2- Pattern Growth Algorithms

1- Apriori-based Graph Mining
2- Frequent Sub-Graph discovery

In these methods, the candidate graph is generated by adding a new edge to the previous candidate. Indeed a frequent subgraph is extended in all manners. The major algorithms in this method are gSpan[3] proposed by Yan et al in 2002, MoFa proposed by Borgelt et al in 2002, FFSM[4] proposed by Huan et al in 2003, SPIN[5] proposed by Huan et al in 2004 and Gaston[6] proposed by Nijssen in 2004 [2]. In gSpan unlike Apriori methods, the search method utilized is DFS [11]. In this method each graph is represented using a DFS canonical code. The isomorphism test is done by comparing these codes. This code is written based on the DFS tree. Since a graph can have many different DFS codes, the code with the conditions noted in [12] is called the minimum DFS code in order to produce a unique code. Two graphs are isomorphic if they have equal minimum DFS codes. If a DFS code is frequent, all of its ancestors are frequent and if not, none of its descendants are frequent. Also if a DFS code is not a minimum code, all of its subtrees could be omitted to reduce the size of the search space.

III. COMPARING ALGORITHMS

Different algorithms of graph mining have different approaches in the fields like graph representation, candidate generation, algorithm approach, frequency evaluation, search strategy, nature of input and completeness of output [6]. In the next sections, we will compare some of these attributes.

3.1. Graph Representation

Graph representation is one the major attributes of an algorithm because it directly affects memory consumption and runtime. Graphs are represented by adjacency matrix, adjacency list, Hash table and Trie data structure [5]. It is notable that the adjacency matrix has numerous empty elements which could waste the time and memory.

3.2. Search Strategy

Search strategy is categorized into DFS and BFS. BFS mines all isomorphic subgraphs, whereas DFS does not do so and therefore, DFS consumes less memory.

3.3. Nature of input

Another classification of the algorithms is based on how they take an input. In the first case, we have a set of graph databases which consist of individual small graphs; Chemical molecules are in this category. In the second case we have a big graph which consists of small subgraphs; Social networks belong to this category.

3.4. Completeness of output

The algorithms can be classified into two groups based on the completeness of the search they carry out and the output they generate. The algorithms in the first group do not mine the complete set of frequent subgraphs, whereas those in the second group discover the whole set of frequent subgraphs. Thus, there is a trade-off between performance and completeness. It is necessary to consider that mining all frequent subgraphs is not always efficient.
Table 1 shows a comparison of the described algorithms based on the explained attributes.

Table 1. comparison between SUBDUE, AGM and gSpan

	SUBDUE	AGM	gSpan
Graph Representation	Adjacency matrix	Adjacency matrix	Adjacency list
Algorithm approach	Greedy search	Apriori based	Pattern growth based
Frequency evaluation	MDL	Canonical form	DFS code
Search strategy	BFS	BFS	DFS
Nature of input	First case	Second case	Second case
Mining all frequent subgraphs	No	Yes	Yes

IV. CONCLUSION

In this paper different graph mining algorithms have been categorized and described. In addition, their differences and structure have been discussed. Because of the importance and efficiency of graphs in different fields of science, graph mining is a rapidly growing field. Graph mining algorithms generally concentrate on frequent substructures. There are various kinds of categorization for algorithms. In brief, SUBDUE is preferred to AGM and gSpan when data is given in large graphs. Logic based systems make more efficient use of background knowledge and are better at learning semantically complex concepts. For graph based methods, the goal is to find those parts of the graph which have more support. These methods guarantee to mine all subgraphs under user-defined conditions. For large databases and those with a high order of stochastic, AGM and gSpan are better choices. gSpan outperforms AGM by an order of magnitude and is capable of mining larger frequent subgraphs in a bigger graph set with lower minimum supports.

REFERENCES

1. Cook, D.J.; Holder, L.B. Graph-Based Data Mining, IEEE Intelligent System, 2000, pp. 32-41.

3- Graph-based Substructure pattern mining
4- Fast Frequent Subgraph Mining
5- SPanning tree based maximal graph mINing
6- GrAph, Sequences and Tree extractiON

2. Han, J.; Pei, J.; Yin, Y. Mining frequent patterns without candidate generation. ACM-SIGMOD International Conference on Management of Data (SIGMOD'00), 2000, pp. 1–12.

3. Inokuchi, A.; Washio, T.; Motoda, H. An apriori-based algorithm for mining frequent substructures from graph data. European Symposium Principle of Data Mining and Knowledge Discovery (PKDD'00), 2000, pp. 13–23.

4. Inokuchi, A.; Washio, T.; Motoda, H. Complete Mining of Frequent Patterns from Graphs: Mining Graph Data. Machine Learning, 2003, pp. 321-354.

5. Kolosovskiy, M.A. Data structure for representing a graph: combination of linked list and hash table. CoRR, 2009.

6. Krishna, V.; Ranga Suri, N. N. R.; Athithan, G. A comparative survey of algorithms for frequent subgraph discovery, Current Science, 2011.

7. Kuramochi, M.; Karypis, G. Frequent subgraph discovery. International Conference on Data Mining (ICDM'01), pp. 313–320, San Jose, CA, Nov. 2001.

8. Matsuda, T.; Horiuchi, T.; Motoda, H.; Washio, T. Extension of graph-based induction for general graph structured data. PAKDD, 2000, pp. 420–431.

9. Muggleton, S. Stochastic logic programs. In L. de Raedt, ed. Advances in Inductive Logic Programming, 1996, IOS Press, Amsterdam, pp.254-264.

10. Vanetik, N.; Gudes, E.; Shimony, S.E. Computing frequent graph patterns from semistructured data. ICDM, 2002, pp. 458–465.

11. Yan, X.; Han, J. CloseGraph: Mining Closed Frequent Graph Patterns. SIGKDD, 2003, pp. 286-295.

12. Yan, X.; Han, J. gSpan: Graph-Based Substructure Pattern Mining. ICDM, 2002, pp. 721-724.