Review of Various Algorithms in Graph Mining Based on Search Strategies

Vijender Singh, Deepak Garg

Abstract- Graph mining is an active research area during these days. Graphs have become significant in the modeling of complicated structures such as circuit images, chemical compounds, protein structures, biological networks, social networks, web workflows and XML documents. A common framework is necessary to study various graph mining algorithms and their applications. In this paper, we present a review study of various algorithms based on their graph representation, subgraph generation, algorithm approach, frequency evaluation and search strategy.

Keywords: Subgraphs, Graph Mining, Algorithms

I. INTRODUCTION

In mathematics, computer science and related subjects an algorithm is an effective method for solving a problem expressed as a finite sequence of instructions. Algorithms are used for calculation in data processing and many others fields. Graphs are the modeling tools to get information from heterogeneous sources. Graph mining tasks in complex systems like chemo informatics and bioinformatics consist of analyzing large collections of molecules with the goal to find some prediction among molecules of a specific class. The molecules can be represented by molecular graphs which are related to graph mining and structured data mining.

A. *Preliminaries:*

A graph is a diagram which consists of collection of vertices together with edges with joining of certain pairs of these vertices. Mathematically, we can write:

A Graph G = [V(G), E(G)]

where the sets V(G) and E(G) are defined as

V(G) = Vertex set of graph G

E(G) = Edge set of graph where each element e of E(G) is assigned an unordered pairs of

vertices (u, v) called end vertices of e.

Subgraph: If G and H are two graph with vertex sets V(H), V(G) and edge sets E(H) and E(G) respectively such that $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$ then we call H as subgraph of G.

II. SURVEY OF THE ALGORITHMS

Various algorithms on graph mining are developed by many researchers. For example, Ullamann [53] in 1976 developed an algorithm for subgraph isomorphism. Subgraph isomorphism is determined by means of a brute-force tree search procedure.

Revised Version Manuscript Received on January 11, 2016.

Vijender Singh, Department of Computer Science and Engineering, Thapar Institute of Engineering and Technology University, Patiala, Punjab 147004 India.

Deepak Garg, Department of Computer Science and Engineering, Thapar Institute of Engineering and Technology University, Patiala, Punjab 147004 India.

This algorithm attains efficiency by inferentially eliminating successor's nodes in the tree search. Agarwal and Srikant [1] in 1994 considered the problem of discovering association rules between items in a large database of sales transaction. They presented two new algorithms for solving this problem that are fundamentally different from the known algorithm. Holder *et al.* [20] in 1994 described the SUBDUE system where the minimum description length (MDL) principle is used to investigate substructures, which helps in compressing the database and representing the structural concepts in the data. Inokuchi, *et al.* [25] in 1998 proposed an AGM algorithm, which mines efficiently the association rule among the frequently appearing substructure in a given graph dataset. Nijssen and Kok [43] in 2004 proposed a "quick start principle" with the fact that the various substructures are contained in each other. For searching the substructures, they considered the first paths and transformed them to trees and finally trees are transformed into the graphs. Following this, one can use more efficient algorithms for the simple substructures where the advanced algorithms are only used in special circumstances. Krishana *et al.* [33] in 2011 presented a comparative survey of algorithm for frequent subgraph discovery, where they classified the intrinsic characteristics of various algorithms. Various algorithms and techniques for finding frequent patterns in graph mining are reviewed by Singh and Garg [51]. In the present paper, we considered 44 algorithms and proposed a common framework for different properties of these algorithms. Some search strategies and approaches of these algorithms are given in section 2 and 3.

III. ASPECTS OF EXISTING ALGORITHMS

The frequent subgraph discovery process is partitioned into two main steps: (a) Finding a subgraph based on search strategy (b) Finding matchings in the given graph using subgraphs isomorphism. Finally, there are following four major aspects of an algorithm which influence the execution and output of the algorithm:

- Graph representation.
- Subgraph generation.
- Algorithm approach.
- Frequency evaluation

Graph representation is an important concept in subgraph discovery algorithm because it has direct and significant influence on memory usage as well as execution time of these algorithms. Various graph representation schemes are available among which adjacency matrix, adjacency list, hash table, M-DFSC (Minimum Depth first search code), trie structure, sparse graph, Edge triplet, canonical adjacency graph, clique detection, three dimensional and two dimensional, spanning tree and path join are frequently used by the graph mining algorithms.

Definition1. Adjacency matrix: if an undirected graph G consists of n vertices then the adjacency matrix of graph is V × V matrix A = [a_{jk}] and defined by

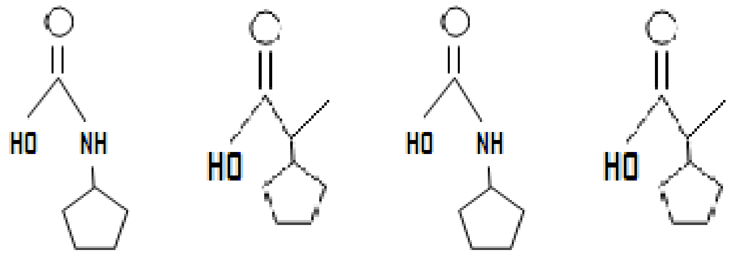
$$a_{jk} = \begin{cases} 1, if \{Vj, Vk\}\ is\ an\ edge\ i.e. Vj\ is\ adjacent\ to\ Vk \\ 0, if\ there\ is\ no\ edge\ between\ Vj\ and\ Vk \end{cases}$$

if there exists an edge between vertex V_j and $V_{k,}$ where j is row and k is column then value of a_{jk} = 1. If there is no edge between vertex V_j and V_j then value of a_{jk} = 0.

Definition2. Adjacency list: The adjacency list representation [6] of a graph G = (V,E) consists of an array Adj of |V| lists, one for each vertex in V. for each e $\in V$, the adjacency list Adj[e] contains all the vertices v such that there is an edge (e, v) $\in E$. i.e. Adj[e] consists of all the vertices adjacent to e in G. The adjacency list representation consumes less space compared to adjacency matrix. For a graph with v nodes, adjacency matrices take O (v^2) space and adjacency list take O (|e|) space. Definition3. Sparse graph: A Graph in which the number of edges is much less than the possible number of edges. Graph G is said to sparse Graph if G = (V,E) in which |E| = O (|V|). For example here we consider a graph G = (V,E) with n nodes suppose that the out-degree of each vertex in G is some fixed constant K. graph G is a sparse graph because |E| = K|V| = O(|V|)

Definition4. Spanning subgraph: If H is a subgraph of G and H contains all the vertices of G.

The hash table schemes use a hash function to map keys with their corresponding values. Hash table is a data structure that uses a hash function to map identifying values. So a hash table implements an associative value. The hash function is used to transform the key into the index of an array element where the corresponding value is to be sought. Advantage of this representation is the speed with which it retrieves the results. This is useful in case of very large input graph. Trie structure is an order tree data structure which can be used to represent graphs. Look up time in trie is less and is comparatively faster than a poorly designed hash table. A trie structure could be a good data structure for building a memory efficient search/dictionary with fast lookup and auto completion. The algorithms like SUBDUE, HSIGRAM, FFSM, Part Graph mining, ASSAM, SyGMA, PATH, GBI, WARMR, Ac GM, McGregor, Page Rank, AGM, Clospan, Prefix Span, MAFIA, SPADE, CODENSE, GIndex, CHARM and Mo Fa use adjacency matrix and canonical adjacency matrix for storing the graph [20; 29; 21; 45; 3,4; 16; 54; 36; 15; 26; 42; 9,48; 25; 59; 46; 10;66; 19; 61; 65; 6]. The algorithms like FSG, gSpan, Close Graph, close-cut, PP-Top & DM-Top K and LCMA, are use as adjacency list and sparse graph [28; 57; 58; 60; 34; 35]. The algorithms FREQT, Cspan, VSIGRAM, g FSG and GREW use as sparse graph for storing the graph [5; 62; 49; 47; 30]. FARMER algorithm is uses as trie structure. [44]. TREEMINER algorithm is uses as Path join. [64]. RASCAL, MCS, MCIS, and MCES algorithms are used as clique and two dimension for representation the graph, spin algorithm prefer spanning tree. MCS is a maximum common subgraph isomorphism algorithm especially designed for graphs obtained from small and largest molecules [50; 11; 12; 18].



(A) Connected MCS **(B) Disconnected MCS**

MolFea algorithm prefers the distribution for storing the graph [31]. INDUCTION and GASTON algorithms used the Hash Table for graph representation and GASTON is found more efficient than other algorithms [63; 43]. The ISG algorithm transforms the input set of graphs into item sets which are then represented by using edge triplets [52].

Subgraph generation is an important aspect in the subgraph discovery process. It has number of mechanism out of which the most common are level-wise search, right most extension, iterative merging, Edge triple extension, Merge join, Edge extension, minimal maximal subgraph, Bipartite matching, sequential pattern, splat edge connectivity, Local clique and clustering coefficient, maximum clique detection, equivalence class and merging. In level wise search, the algorithm finds a subgraph and then enumerates the instances of the subgraph by one adjacent edge in all possible way. Extension is an already discovered subgraph is extended by one edge or node at a time. Rightmost extension is basic methodology is to extend the graph rightmost vertex or rightmost path of graph. Iterative merge has a bottom-up approach. It starts by the smallest block size where it compares with the 1st entry with the 2nd entry, 3rd entry with 4th entry and so on. After 1st pass the block size increase to twice previous size. It repeats the same iteration until the block size reaches the length of the adjacent element is merged to form sorted pair. This adjacent pair is merged to form 4 tuples and so on. This is an O(nlogn) algorithm. Edge triplet extension is a discovered itemset is extended by adding one edge triplet in which iteration. Merging is a subgraph discovered in previous iteration and connected by one or more edges are merged to obtain a new graph. Minimal maximal subgraph (MMS) is matching in a graph is a set of edges without common node. Given a graph G = (V, E) a matching M in G is a set of pair wise non-adjacent edges that is no two edges share a common vertex. A vertex is matched if it is an endpoint of one of the edges in the matching. A Maximal subgraph matching is matching that contain largest possible number of edges while minimal matching graph is matching that contain minimum possible number of edges. A perfect matching is also a minimum size edge cover. There are two approaches to mining such closed dense graphs efficiently; a pattern-growth approach called close cut and a pattern-reduction approach called splat. Given a graph G, an edge cut is a set of edges E_1 such that E (G)-E_1 is disconnected. A minimum cut is the smallest set in all edge cuts. The edge connectivity of G, written K (G) is the size of a minimum cut. A clustering coefficient is a measure of degree to which nodes in graph tend to cluster together. Local clustering coefficient gives an indication of the embedded of single node. The local coefficient of a vertex in graph quantifies how close its adjacent is to being a clique. While processing an execution trace we create equivalence class of execution edge. On encountering a new edge leading up to waiting time event, we compare the edge to the set of equivalence classes. If the edge does not match any existing equivalence class, a new class is created with that edge as both the representative of the class and the current best characterization for the class, subsequent edge are compared to representation for each class to see if they belong in the same equivalence class. If an edge matches the representation, it merged with the characterization for that equivalence class.

In level-wise search, the algorithms find a subgraph and enumerate the instances of the subgraph by one adjacent edge in all possible ways. SUBDUE, MolFea, GBI, WARMR, AcGM, Grafil, GIndex and FARMER follows this mechanism for subgraph generation [20; 31; 36; 15; 26; 60; 61; 44]. The mechanism of edge extension where is an already discovered subgraph is extended by one edge and node at a time. FSG algorithm employed this method. Whereas gSpan, closeGraph, close-cut, clospan, Prefix span, MAFIA, CHARM, PP-TopK & DM-Top K and FREQT used a modified version of extension which is called rightmost extension [28; 57; 58; 62; 59; 46; 10; 65; 35; 5] . In rightmost extension, the basic methodology is to extend the graph from the rightmost vertex or from right most path of the graph. ISG uses a similar approach which is known as edge triplet extension in this item set is extended by adding one edge triplet. The third common mechanism is merging in which certain subgraph which connected by one or more edges are merged to certain a new graph. GREW, HSIGRAM, VASIGRAM used merging subgraph generation. FFSM used a combination of merging and extension for subgraph generation. GASTON used a combination of cycle closed refinement and NAUTY-based normalization [40] cycle close refinement ensures that graph are generated only from their corresponding spanning trees. Subgraph generation of the remaining algorithms depending on various other mechanism are given in Table 1-3.

***Algorithm approaches* are of three types as given below**

(i) Apriori – based Approach

Apriori – based frequent substructure mining algorithm share similar characteristics with Apriori-based frequent item set mining algorithms. The search for frequent groups starts with graphs a small “size” and proceeds in a bottom-up manner by generating candidate having an extra vertex, edge or path. The definition of graph size depends on algorithm used. AGM, AcGM, FARMER, FSG, FFSM, FREQT, gFSG, gIndex, ISG, MolFea, Part Graph Mining, PATH, TREEMINER, and VSIGRAM algorithms are following this approach [25; 26; 44; 28; 21; 5; 47; 61; 52; 31; 45; 54; 64; 49].

The Apriori-based approach has to use the breadth-first search (BFS) strategy because of its level-wise candidate generation. In order to determine whether a size (k+1) is frequent. Apriori graph adopts a level-wise mining methodology.

(ii) Pattern – growth Approach:

SUBDUE, gSpan, CloseGraph, HSIGRAM, GREW, GASTON, MoFa, ASSAM, SyGMA, SPIN, Close-Cut, Page Rank, Clospan, Prefix Span, MAFIA, CHARM, SPADE, Cspan, and CODENSE algorithms are follow pattern growth approach.

Pattern growth is more flexible regarding its search method in like DFS which consume less memeory.

Two size K frequent graphs are joined only if they have same size (k-1) subgraph

(iii) Approximate Approach

In this pattern [13] it generates to mine approximate frequent substructures, which allow tiny substructure variations. By this structure can represent several tinny different frequent substructures using one approximate substructure. SUBDUE, MCS, MCES, MCIS, GBI, WARMR, Mcgregor, PP-TopK & DM-TopK, LCMA, grafil and RASCAL algorithms follow in approximation and clique approach methods.

Inductive Logic programming (ILP): Inductive logic programming (ILP also subfield of machine learning) is used in logic programming as a uniform representation. Given an encoding of the known background knowledge and a set of example represented as a logic database of facts, an ILP system will derive a hypothesized logic program which entails the entire positive and none of the negative examples. Inductive logic programming is useful in bioinformatics and natural language processing. Inductive logic programming approach is based on a combination apriori based. FARMER algorithm follows this category.

Schema: Positive Example + negative example + background knowledge = hypothesis.

Frequency evaluations are the process of counting the number of occurrence of a subgraph in the large graph database and determine whether it is frequent or not. A Subgraph is frequent if its count is greater than a predefined threshold value. Frequency evaluation is using many techniques like MDL (minimum description length) code used in SUBDUE. Minimum description Length (MDL): Adriaans and Zantingae [2] stated that the best theory to explain a set of data is the one that minimizes the sum of the length in bits, of the description of the theory and the length in bits of the data when encoded with help of the theory. It is a strategy to help us make a selection between different solutions of problems. Trie data structure is used in FARMER algorithm. Transaction identifier (TID) list is used for frequency evaluation in FSG and ISG algorithms. Each subgraph has a list of transaction identifier which supports it. For enumerating the frequency of H subgraph the intersection of the

Transaction identifier list of (H-1). Subgraph is enumerated close graph, Gspan, MolFea, Page Rank and gIndex algorithms are use DFS lexicographic order for frequency evaluation. Definition 5 Lexicographic order: Lexicographic order is an order function a way of sorting information. It is generally a simple and useful way to sort string. The name comes from the order used in a dictionary where string is compared in alphabetical order from left to right.

Let (X, ≤) be poset (a partial ordered set). We may order the Cartesian product X × X by (a, b) ≤ (c, d) if a ≤ c or a = c, b ≤ d. Then X × X becomes a partially order set under this relation, which is called Lexicographic order.

Minimum DFS code obtained from this for a particular graph is the canonical label of that graph which helps in frequency evaluation. HSIGRAM, GREW and other (which given in table 1) algorithms are to find the maximal independent set of a graph which is constructed out of the embeddings of a subgraph for frequency evaluation. A maximal independent set is an independent set that is not a subset of any other independent set. It is a set G such that every edge of the graph has at least one endpoint not in G and every vertex not in G. it has at least one adjacent in G. A maximal independent set is also a dominating set i.e. independent must be maximal independent. A largest maximal independent set is known as a maximum independent set. FFSM and others algorithms used a sub-optimal canonical adjacency matrix tree for evaluating the frequency. Embedding lists are counting the frequency in GASTON and other's algorithms (shown in table1-3). Meinl et.al. [41] defined that an embedding is a subgraph isomorphism of a subgraph in the lattice to a graph in the database. The embedding list contains all possible embeddings. In the embedding list the subgraph isomorphism's are stored whereas in the appearance list only the molecules are mentioned. A subgraph appears in embedding list is very helpful in the frequency calculation.

The classification of graph mining algorithms are based on search strategy, input and output as given in Figure 1.

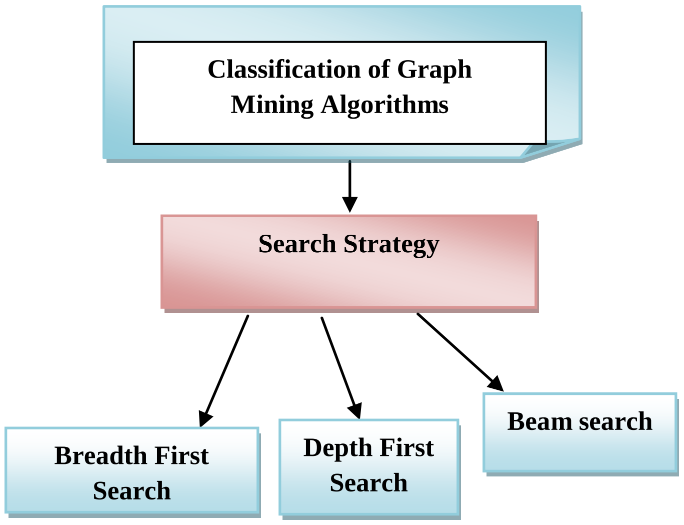


Figure 1. Classification of Graph Mining.

We can classify various Graphs mining algorithm on the basis of search strategy. The search strategy is of three types: BFS, DFS, and Beam search.

(i) Breadth first search (BFS): Breadth first search is begins at the root node and explores all the neighboring nodes. Then for each of those nearest nodes. It explores their unexplored neighbor node and so on until it finds the goal. Based on this strategy, ten algorithms are presented in Table 1.

Table1. Various Aspects of Algorithms based on BFS

Algorithm	Graph Representation	Sub graph generation	Approach	Frequency evaluation	Nature of Input	Completeness of output
SUBDUE	Adjacency matrix	Level-wise search	Approximate	MDL & Background knowledge	Single large graph	Incomplete
FARMER	Trie structure	Level-wise search	ILP & Apriori based	Trie data structure	A set of small graph	Complete
FSG	Adjacency list & sparse graph representation	One edge extension	Apriori based	Transaction Identifier (TID)	A set of small graph	Complete
HSIGRAM	Adjacency matrix	Iterated merging	Apriori Based	Maximal independent set	Single large graph	Complete
PATH	Adjacency matrix	Iterative merging	Apriori-based	Maximal	n/a	n/a
SPIN (spanning tree based maximal graph mining	Spanning tree	Minimum maximal subgraph	Pattern growth	Embedding of graph in a graph database	Large set of graph	Complete
WARMR	Adjacency matrix	Level-wise	Approximate	ILP	n/a	n/a
Ac GM	Adjacency matrix	Level-wise	Apriori	Level-wise join	n/a	N/A
AGM	Adjacency matrix	Level-wise	Apriori	Level-wise join	n/a	n/a
CHARM	Canonical adjacency matrix	Rightmost extension	Pattern growth	Transaction list	Small set graph	Complete
SPADE	adjacency matrix	Sequencial pattern	Apriori growth	Transaction list	Small set graph	Complete

(ii) Depth first search (DFS): According to Deo [17] Depth first search is a powerful technique of systemically traversing the edges of a given graph such that every edges are traversed exact by once and each vertex is visited at least once. This technique is called DFS or backtracking. Thirty one algorithm based on DFS Strategy are given in Table 2.

Table 2. Various aspects of algorithms based on DFS

Algorithm	Graph Representation	Sub graph generation	Approach	Frequency evaluation	Nature of Input	Completeness of output
gSpan	Adjacency list & sparse graph representation	Rightmost extension	Pattern growth	DFS lexographic order	A set of small graph	Complete
Close graph	Adjacency list & sparse graph representation	Rightmost extension	Pattern growth	DFS lexographic order	A set of small graph	Incomplete
GREW	Sparse graph representation	Iterative merging	Pattern growth	Maximal independent set	Single large graph	Incomplete
FFSM	Adjacency matrix	Merging & extension	Apriori-based	Suboptimal canonical adjacency matrix tree	A set of small graph	Complete
Gaston	Hash table	Extension	Pattern growth	Embedding list	A set of small graph	Complete
ISG	Edge triplet	Edge triplet extension	Apriori based	TID list	A set of small graph	Incomplete
MoFa	Canonical graph	Extension	Pattern growth	Embedding list	Small set of graph	Complete
MCS	Adjacency & clique detection	Isomorphism between two graph	Heuristic on clique based	Lexographic & clustering	Large set of graph	Complete
MCES	Adjacency & labeled	Edge based	Heuristic or clique based	Clustering	Large set of graph	Incomplete
MCIS	3D and 2D and isomorphic	Vertex based	Approximate	Clique based induction	Large set of graph	n\a
Gfsg	Sparse graph representation	Extension level-wise	Apriori	Counting framework	A set of small graph	Complete
MoLFEA	Distribution of class	Level-wise	Apriori	ILP & lexographic	Large set of graph	Complete
PART GRAPH MINING	Adjacency matrix	Merge join	Apriori	Clustering	Large graph set	complete
ASSAM	Adjacency matrix	Subgraph matching	Pattern growth	3D space matching isomorphism	Small set of graph	Complete
SyGMA (symmetry free graphmining Algorithm	Adjacency matrix	Edge extension	Pattern growth	Lexographic	Large set of graph	Complete
SPIN(spanning tree based maximal graph mining	Spanning tree	Minimum maximal subgraph	Pattern growth	Embedding of graph in a graph database	Large set of graph	Complete
Close cut	M-DFSC & Adjacency list	Rightmost extension	Pattern growth	Transaction list	Single set graph	Complete
McGregor	Adjacency matrix & clique	Isomorphism	Clique	Backtracking search	Large set of graph	Complete
PageRank	Adjacency matrix	Bipartite matching	Pattern growth	Lexographic order	N/A	N/A
Induction	Hash table	Extension	Apriori based	Embedding	Large set of graph	complete
Clospan	Adjacency matrix	Right most extension	Pattern growth	Lattice & compact transaction list	Small set of graph	complete
PrefixSpan	Canonical adjacency matrix	Rightmost extension	Pattern growth	Transaction list	Small set graph	complete
MAFIA	adjacency matrix	Rightmost extension	Pattern growth	Transaction list and clustering	Small set graph	Complete
VSIGRAM	Sparse graph & adjacency matrix	low level recursively & extensi	Apriori	Embedding	Singal large graph	Complete
Cspan	Sparse graph representation	Splat edge connectivity	Pattern growth	Condensation	Small set of graph	Complete
CODENSE	adjacency matrix	Sparse graph	Pattern growth	Clustering& minimum cut	Small set graph	Complete
PP-TopK & DM-TOP K	Adjacency list	Right most extension	Approximate pattern	Jointly with max. entropy	Large set of graph	Complete
GIndex	Adjacency matrix & hash table	Level-wise indexing	Apriori	Lexographic order & embedding list path based	Small set of graph	incomplete
RASCAL(rapid similarity calculation)	2D graph metching	Max. clique detection matching with screen rigoursely	Approximate	Clustering	Small set of graph	Complete
TREEMINER	Path join	Equivalence class extension	Apriori	Scope list join	n/a	n/a
FREQT	Sparse graph & labeled matrix	Right most path extension	Apriori	Occurrence list	n/a	n/a

(iii) Beam search: Beam search determine the maximum number of substructures that are retained for expansion in the next iteration of discovery algorithm. Beam search is also known as heuristic search. A study on four algorithm based on Beam search Strategy is presented in Table 3.

Table 3. Various aspects of algorithms based on Beam Search

Algorithm	Graph Representation	Sub graph generation	Approach	Frequency evaluation	Nature of Input	Completeness of output
SUBDUE	Adjacency matrix	Level-wise search	Approximate	MDL & Background knowledge	Single large graph	Incomplete
GBI (graph based induction)	Adjacency matrix	Level-wise	Approximate	ILP	Large graph set	Complete
LCMA (Local clique merging algorithm)	Adjacency list	Local clique & clustering coefficient	Approximate	Merging all subgraph & multiple local clique	Single large graph set	Complete MIPS 6.38
Grafil (graph similarity filtering)	Canonical adjacency matrix	Level-wise filtering	Approximate	Hierarchical agglomerative clustering & embedding feature	large set graph	Complete

IV. DISCUSSION

The algorithms given in Tables 1 to 3 are distinguished on the basis of search criteria, BFS, DFS and Beam Search, respectively. This classification scheme enables the developers to choose an appropriate algorithm for a particular applications depending on the resource constraints or environment.

(a) Based on BFS search strategy

In Table 1, we have classified the various algorithms based on BFS search strategy. The advantage of BFS strategy is that it provides all frequent subgraphs above a minimum threshold given relaxed memory and time constraints. This is due to the fact that BFS is a level by level search, which considers all nodes at a particular level and searches all the nodes in the graph. It has low vulnerability to redundancy as compared to DFS. The main drawback with algorithms based on BFS given in Table 1 is that they are slower than their counterparts that use the DFS strategy. However, these algorithms are popular among the frequent subgraph discovery community. SUBDUE is the earliest algorithm that implemented in beam search or BFS. Other algorithms like GBI, LCMA, Grafil, SPADE, CHARM, AGM, AcGM, WARMR, PATH, HSIGRAM, and FARMR are also fall in this category.

(b) Based on DFS search strategy

The algorithms based on DFS search strategy as given in Table 2 consumes less space compared to BFS. This is due to the fact that the number of lists that need to be stored in memory in case of DFS is proportional to the depth of the graph. Due to this reason, most of the recent algorithms attracted towards DFS approach. However, the disadvantage of DFS approach is high probability of redundancy generated subgraphs. But gSpan is the first algorithm implemented using the depth first search strategy. CloseGraph, Grew, FFSM, GASTON, ISG, MoFa, MCS, MCES, MCIS, gFSG, MolFea, Part Gaph Mining, ASSAM, SyGMA, SPIN, Close-Cut, McGregor, PageRank, Induction, CloSpan, PrefixSpan, MAFIA, VSIGRAM, Cspan, CODENSE, PP-TopK & DM-TopK, gIndex, RASCAL, TREEMINER and FREQT algorithms also follow the same search strategy. GASTON is most efficient algorithm among the lot of and its efficiency is mainly due to its depth first search strategy.

(c) Based on Beam Search strategy

In computer Science beam search is a heuristic search strategy algorithm that explores a graph by expanding the most promising node in a limited set. Beam search is an optimization of best-first search strategy that reduces its memory requirements. Best first search is an order of all partial solution according to some heuristic which attempts to predict how close a partial solution is to a complete solution. Beam search uses breadth-first search to build its search tree. The extension of beam search has been made completely by combining it with DFS resulting in beam stack search and depth first beam search and limited discrepancy. The algorithms SUBDUE, LCMA, GBI and Grafil given in Table 3 are based on Beam Search strategy. These algorithms use adjaceny matrix/adjaceny list for their graph representation. Like BFS, they are also slower than their counter parts that use the DFS strategy.

V. CONCLUSION AND FUTURE SCOPE

Various graph mining algorithms are classified according to search strategies. The advantages and disadvantages of these algorithms are discussed. The algorithms for frequent pattern mining become very costly in time and space as the pattern sizes and network number increase. Currently no efficient algorithm is available for mining recurrent patterns across large collection of genome wide network. There are various domains like chemoinformatics bioinformatics etc. where no efficient algorithms are available, for example, for mining recurrent patterns across large collection of genome-wide networks. Due to increasing size and complexity of patterns in computer sciences the need for efficient graph mining algorithm is increasing. Still there is a scope of improvement in graph mining algorithm; the improvement can be in speed or sensitivity.

REFERENCES

1. Agarwal, R. Srikant, R. Fast algorithm for mining association rules. In the proc. Of the 20th Int. conf. on very large databases (VLDB),1994, 487-499..
2. Adriaans, P. Zantinge, D. Data Mining. Pearson education Asia, 2002.

3. Artymiuk, P.J. Spriggs, R.V. Willett, P. Graph theoretic methods for the analysis of structural relationships in biological macromolecules. Journal of the American society for information science & technology, 2005, 56:5, 518-528.
4. Artymiuk, P.J. Poirrette, A.R. Grindley, H.M. Rice, D.W. Willett, P. A graph-theoretic approach to the identification of three dimensional patterns of amino acid side-chains in protein structures. Journal of molecular biology, 1994, 243, 327-344.
5. Asai, T. Abe, K. Kawasoe, S. Arimura, H. Satamoto, H. Arikawa, S. Efficient substructure discovery from large semi-structured data. In proceedings for the 2002 SIAM international conference on data mining (SDM'02) Arlington VA, 2002, 158-174.
6. Borgett, C. On canonical forms for frequent Graph Mining. Springer, 2007, 337-349.
7. Borgelt, C. Berthold, M.R. Mining molecular fragments: Finding relevant substructures of molecules. In proc. 2002 int. conf. Data mining(ICDM'02), IEEE, 2002, 51-58.
8. Borgelt, C. Meinl, T. Berthold, M.R. Advanced Pruning strategies to speed up mining closed molecular fragments. IEEE (int. conf.), 2004, 4565-4570.
9. Brin, S. Page, L. The Anatomy of a large-scale hypertextual web search engine. In proc. 7th International conference on the www., 1998, 107-117.
10. Burdick, D. Calimlim, M. Gehrke, J. MAFIA: A maximal frequent itemset algorithm for transactional databases. IEEE, 2005, 17:11, 1490-1504.
11. Conte, D. Guidobaldi, C.Sansonr, C. A comparison of three maximum common subgraph algorithms on a large database of labeled graphs. IAPR workshop GBPR 2003, LNCS, 2003, 2726, 130-141.
12. Cao, Y. Jiang, T. Girke, T. A maximum common substructure-based algorithm for searching and predicting drug like compounds. Bioinformatics, 2008, 24:13, i366-i374.
13. Deshpande, M. Kuramochi, M. Karypis, G. Automated approaches for classifying structures. In Proc. 2002 workshop on Data mining in Bioinformatics (BIOKDD'02), 2002, 11-18.
14. Dehaspe, L. Toivonen, H. King, R.D. Finding Frequent substructures in chemical compounds. In 4th international conference on knowledge discovery and data mining. 1998.
15. Dehaspe, L. Toivonen, H. Discovery of frequent DATALOG Patterns. Data mining and knowledge discovery, 1999, 3, 7-36.
16. Desrosiers, C. Galinier, P. Hertz, A. Improving frequent subgraph mining in the presence of symmetry. 5th international workshop on mining, 2007.
17. Deo, N.S. Graph theory with applications to engineering and computer science. Prentice hall of India new Delhi, 2008
18. Ehrlich, H.C. Rarey, M. Maximum Common Subgraph isomorphism algorithms and their applications in molecular science: a review. John wiley and sons ltd, 2011, 1, 68-79.
19. Hu, H. Yan, X. Huang, Y. Han, J. Zhou, X.J. Mining coherent dense subgraphs across massive biological networks for functional discovery, Bioinformatics,2005, 21, i213-i221.
20. Holder, L.B. Cook, D.J. Djoko, S. Substructure Discovery in the SUBDUE system. In proceedings AAAI'94 workshop knowledge Discovery in Database (KDD'94),1994, 169-180.
21. Huan, J. Wang, W. Prins, J. Efficient Mining of frequent subgraph in the presence of Isomorphism. In proc. 2003 Int. conf. Data mining (ICDM'03), 2003, 549-552.
22. Helal, N.A. Soliman, T.H.A. Karam, O.H. Analyzing efficient embedding list structures in subgraph mining. IADIS European conference data mining, 2007, 129-134.
23. Haun, J. Wang, W. Prins, J. Yang, J. Spin: Mining Maximal Frequent subgraph from graph databases. KDD'04 Seattle, Washington, USA. 2004.
24. Han, J. Kamber, M. Data Mining: Concepts and techniques. Second edition, Elsevier, 2006.
25. Inokuchi, A. Washio, T. Motoda, H. An Apriori-based Algorithm for Mining Frequent Substructures from Graph Data. In Proc. 2000 European Symp. Principle of Data mining and Knowledge Discovery (PKDD'00), 1998, 13-23.
26. Inokuchi, A. Washio, T. Nishimura, K. Motoda, H A fast algorithm for mining frequent connected, research report IBM Japan, 2002
27. Jiang, C. Coenen, F. Zito, M. Frequent Sub-graph on Edge Weighted Graphs. Springer, 77-88, 2010.
28. Kuramochi, M. Karypis, G. Frequent Subgraph Discovery. In Proc. 2001 Int. conf. Data mining (ICDM'01), 313-320, 2001.
29. Kuramochi, M. Karypis, G. Finding frequent patterns in a large sparse graph. Data mining knowledge discovery, 11(3), 243-271, 2005.
30. Kuramochi, M. Karypis, G. GREW a scalable frequent subgraph discovery algorithm. Technical report 04-024, University of Minnesota, department of computer science, 2004.
31. Kramer, S. Raedt, L.D. Helma, C. Molecular Feature mining in HIV data. In Proceedings of the seventh ACM SIGKDD International conf. on knowledge discovery and data mining, pp. 136-143, 2001.
32. Koyuturk, M. Grama, A. Szpankowski, W. An efficient algorithm for detecting frequent subgraphs in biological networks. Bioinformatics, 20: i200-i207, 2004.
33. Krishna, V. Suri, N.R.R.R. Athithan, G. A comparative survey of algorithms for frequent subgraph discovery. Current science, 100:2, 190-198, 2011.
34. Liu, Y. Li, J. Zhu, J. Gao, H. Mining Top-K graph patterns that Jointly maximize some significance Measure. J. of computers, 5(4), 566-572, 2010.
35. Li, X.L. Tan, S.H. Foo, C.I. Ng, S.K. Interaction graph mining for protein complexes using local clique merging. Genome Informatics, 16(2): 260-269, 2005.
36. Matsuda, T. Horiuchi, T. Motoda, H. Washio, T. Extension of graph-based induction for general graph structured data. In Proceeding of 4th Pacific Asia conference of knowledge discovery and data mining (PAKDD 2000),2000, 420-431.
37. Meinl, T. Borgelt, C. Berthold, M.R. Discriminative closed Fragment Mining and perfect extensions in MoFa. 2004.
38. Meinl, T. Worlein, M. Fischer, I. Philippsen, M. Mining Molecular datasets on Symmetric Multiprocessor systems. 2006.
39. Meinl, T. Worlein, M. Urzova, O. Fischer, I. Philippsen, M. The ParMol package for frequent subgraph mining. Electronics communication of ESST 1, 2006.
40. McKay, B.D. NAUTY. Users guide (version 1.5) technical report, TR-CS-90-02, Department of computer science, Australian National University, 1990.
41. Muggleton, S. Inductive logic Programming, Academic Press, 1992.
42. McGregor, J. backtrack search algorithms and the maximal common subgraph problem. Software practice and experience,1982, 12, 23-34.
43. Nijssen, S. Kok, J.N. A Quickstart in frequent structure mining can make a difference. In proceedings of 10th ACM SIGKDD International conference on knowledge Discovery and Data mining, ACM,2004, 647-652.
44. Nijssen, S. Kok, J.N. Faster association rules rules for multiple relations. In IJCAT'01: 17th Int. joint conf. artificial intelligence, 2001, 2, 891-896.
45. Nguyen, S.N. Orlowska, M.E. Li, X. Graph mining based a data partitioning approach. In proceedings of the 19th conference on Australasian database, 2008,75, 31-37.
46. Pei, J. Han, J. Mortazavi-Asl, B. Pinto, H. PrefixSpan: Mining Sequential Patterns efficiently by prefix-Projected Pattern Growth. In Proc. 2001 Int. conf. Data Engineering (ICDE'01), 2001, 215-224.
47. Priyadarshini, S. Mishra, D. g-FSG Approach for finding subgraph. IJCCT, 2010,1:2,3,4, 68-71.
48. Page, L. Brin, S. Motwani, R. Winograd, T. The page rank citation ranking: Bringing order to the web. TR 1999-66 stanford university Stanford USA, MA 1998.
49. Reinhardt, S. Karypis, G. A multilevel parallel implementation of a program for finding frequent patterns in a large sparse graph. IEEE, 2007.
50. Raymond J.W. Gardiner, E.J. Willett, P. RASCAL: calculation of graph similarity using maximum common edge subgraphs. The computer journal, 2002, 45:6, 631-644.
51. Singh, V. Garg, D. "Survey Of finding frequent patterns in Graph Mining: Algorithms and Techniques" IJSCE, 2011, 1:3,19-23.
52. Thomus, L. Valluri, S. Karlapalem, K. Isg: Itemset based subgraph mining. Technical report, IIIT, Hyderabad, 2009
53. Ullamann, J.R. "An algorithm for subgraph isomorphism". J. ACM, 23, 1976, pp. 31-42.
54. Vanetik, N. Gudes, E. Shimony, S.E. computing frequent graph patterns from semistructured data. In proceedings of 2002 IEEE international conference on data mining (KDM), 2002,458-465.
55. Worlein, M. Meinl, T. Fischer, I. Philippsen, M. A Quantative comparison of the subgraph miners MoFa, gSpan, FFSM and GASTON. 2005.
56. Washio, T. Motoda, H. State of the art of Graph-based data mining. SIGKDD Explorations,2003, 5:59-68.
57. Yan, X. Han, J. gSpan: Graph-Based Substructure Pattern Mining. In Proc. 2002 Int. conf. Data mining, 2002, 721-724.
58. Yan, X. Han, J. CloseGraphs : Mining Closed Frequent Graphs Patterns. In Proc. 2003 ACM SIGKDD Int. conf. knowledge Discovery and Data mining (KDD'03), 2003, 286-295.

59. Yan, X. Han, J. Afshar, R. CloSpan: Mining Closed Sequential Patterns in Large Datasets. In Proc. 2003 SIAM Int. conf. Data mining (SDM'03), 2003,166-177.
60. Yan, X. Yu, P.S. Han, J. Substructure similarity search in graph databases. In Proc. 2005 ACM-SIGMOD Int. conf. Management of Data (SIGMOD'05), 2005, 766-777.
61. Yan, X. Yu, P.S. Han, J. Graph Indexing: A frequent structure-based approach. In Proc. 2004 ACM-SIGMOD Int. conf. Management of Data (SIGMOD'04), 2004.
62. Yan, X. Zhou, X.J. Han, J. Mining closed relational graphs with connectivity constraints. ACM, (KDD05), 2005.
63. Yoshida, K. Motoda, H. Indurkhya, N. graph based induction as a unifield learning framework. J. of applied of Intel,1994, 4:297-328.
64. Zaki, M.J. Efficiently Mining Frequent trees in a forest. In Proc. 2002 ACM SIGKDD Int. conf. knowledge Discovery and Data mining (KDD'02), 2002, 71-80.
65. Zaki, M.J. Hsiao, C.J. CHARM: an efficient algorithm for closed itemset mining. In proceeding of the 2002 SIAM international conference on data mining (SDM'02) Arlington, VA, 2002, 457-473.
66. Zaki, M.J. SPADE: an efficient algorithm for mining frequent sequences. Mach. Learn., 2001,40, 31-60.