International Journal of Engineering & Technology, 7 (3.8) (2018) 58-62

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

A Survey of Frequent Subgraph Mining Algorithms

Chandra Prakash Dixit[1]*, Nilay Khare[2]

[1] *Research Scholar, Maulana Azad National Institute of Technology, Bhopal, M.P., India*
[2] *Associate Professor, Maulana Azad National Institute of Technology, Bhopal, M.P., India*
** Corresponding author E-mail: dixitcpd@gmail.com*

Abstract

Graphs are broadly used data structure. Graphs are very useful in representing/analyzing and processing real world data. Evolving graphs are graphs which are frequently changing in nature. There is either increase or decrease in their size i.e. change in number of edges or/and vertices. Mining is the process done for knowledge discovery in graphs. Detecting specific patterns with their number of repetition more than a predefined threshold in graph is known as frequent subgraph mining or FSM. Real Timed data representing graphs are high volumetric or of very large in size, handling such graphs require processing them with special mechanisms and algorithms. Our review paper discovers present FSM techniques and tries to give their comparative study.

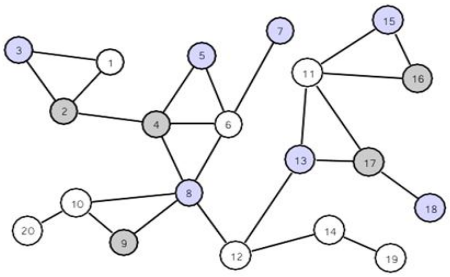
Keywords: *Data Mining; Frequent Subgraph Mining; Graph Algorithms; Indexing; Knowledge Data Discovery; Pattern Mining;*

1. Introduction

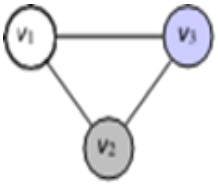
Researchers have worked in recent years on converting real time data into meaningful format aiming to process or analyze it easily. Among those useful and broadly used format, graph is one. Frequent subgraph mining i.e. FSM is a vital and broadly studied graph process which has application in lots of domains e.g. Bioinformatics, Cheminformatics, Defense and Security, Social Networks, Data Mining etc. and that's why FSM has been a center of interest of many researchers. It is also termed by scholars as Frequent Subgraph Discovery or Frequent Pattern Mining. (Figure 1)

Modern graph datasets can be easily understood with two words- 'Large and Evolving', By saying them Large we mean number of nodes and edges present in that graph are in multiples of millions and billions, and term Evolving mean these graphs aren't static frequently changing with addition and/or deletion of nodes and/or edges. A considerable problem with evolving or dynamic graphs is that an insertion may result in one or more subgraphs which are not our targeted subgraph, becoming our targeted subgraph pattern and one single deletion may result in conversion of one or more currently targeted subgraph present in current graph into non targeted subgraph or subgraph of our targeted subgraph. Modern graph datasets may seem easy to describe but these are equally complex to analyze/ process and extract information from these, and these facts makes FSM more interesting and an area with scope as vast as sky and as far as horizon.

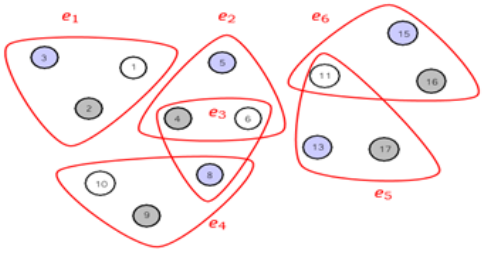
Challenges in FSM doesn't only ends with its being "Large and Evolving". Another challenge comes in form of graph data being wrapped in one unit or split in smaller multiple units, i.e. Graph is single connected component or disconnected and is in smaller units. These two types of graphs give different performance with various algorithms. That's why choosing a right algorithm for type of data set used is also very important.



: (a) Sample Graph



(b) Pattern



(c) Possible Subgraphs

Figure 1

In this Review Paper we will mainly study and compare existing works on Subgraph Mining. Thus we hereby present a comparative survey on Frequent Subgraph Mining (now on referred as FSM) and FSM Algorithms. Then, we will proceed to identify

present graph processing frameworks, then we'll highlight various existing solutions in various types of graphs i.e. large and single unit graph or small sized and disconnected graphs. With the help of literature survey we will be able to identify various varieties of subgraph and graph mining and graph knowledge discovery and exploration techniques w.r.t format.

2. Graph Definitions

Definition 2.1: *A* **Graph** *defined as G = (V, E) consist of set of vertices V and set of edges E. An edge* (v_i, v_j) *where* $\{v_i, v_j \in V\}$ *is also defined as pair of end vertices between of which that particular edge is present.*

Definition 2.2: Labeled Graph *is a precise type of graph where every vertex and edge has a tag or label associated with it. Labels of Vertices is drawn from a set of labels named as VertexLabel(*L_V*) similarly Labels of edges is drawn from a set of labels named as EdgeLabel(*L_E*).*

Definition 2.3: Unlabeled Graph *is a graph in which all vertices are wither assigned same labels or no label is assigned to them. Given a graph* $G_{main} = (V_{main}, E_{main})$*, a graph* $G_{sub} = (V_{sub}, E_{sub})$ *is a subgraph of* G_{main} *if and only if* $V_{sub} \subseteq V_{main}$ *and* $E_{sub} \subseteq E_{main}$.

Definition 2.4: *A graph is* **connected** *if one or more path exist between every vertex pair in the graph else graph is* **disconnected graph**. *Connected graph is always Uni-component graph i.e. has only one component.*

Definition 2.5: *Two graphs* $G_x = (V_x, E_x)$ *and* $G_y = (V_y, E_y)$ *are* **isomorphic** *if they are topologically identical to each other, i.e., there is a mapping from* V_x *to* V_y *in such a way that each edge in* E_x *is mapped to a single edge in* E_y *and vice versa.*

Definition 2.6: *Given two graphs* $G_x = (V_x, E_x)$ *and* $G_y = (V_y, E_y)$*, the problem of* **subgraph matching** *or* **subgraph isomorphism** *is to find an isomorphism between* G_y *and 'a subgraph of* G_x*', i.e., to determine whether* G_y *is a part of* G_x *or not. In labeled graphs, isomorphic mapping must preserve labels on the edges and vertices also.*

3. Overview of FSM

This section of our survey is to give an outline of FSM mining process. FSM mining process consists of 3 aspects.

3.1. Graph Representations

Graph representation aspect of FSM is a mechanism with which a graph can be represented. Simplest Graph representation mechanism is adjacency list or adjacency matrix. In adjacency Matrix representation, rows and columns of that matrix represent a node or vertex. A positive intersection of ith row and jth columns tells us that there is an edge present connecting V_i and V_j. In easy words value $<V_i, V_j>$ gives number of edges present between V_i & V_j, i.e. if its 0 then V_i & V_j aren't connected else if the value is *n* then *n* links or edges are present between these pair of two vertices. Since Graphs can be represented in many diverse ways so for adjacency list it's dreadful to detect isomorphism. This weakness can be overcome by canonical naming because it derives unique code for each graph and this strategy ensures that two identical graphs are labeled similarly thus it make sure that canonical labels of two or more graphs are identical iff those graphs are isomorphic.

3.2. Subgraph Enumeration

Enumeration is process of calculate/count something. Counting subgraphs one by one is subgraph enumeration. There are currently two categories of graph enumeration

1) *Join Operation, and*
2) Extension operation.

Both the operations have few concerns associated with them like, in single join operation multiple candidates can be proposed by single join operation while by many join operations may propose a candidate which might be redundantly. While considering Extension operation, it might restrict nodes which are attached to newly introduced edge.

3.3. Frequency Counting

Graph counting is done in two ways Embedding Lists (EL), and Recomputed Embedding. Single node graph is stored in an EL as list of every label occurrence in database while other graphs are kept in EL. "This list contains *i)* index of embedding tuple in EL of predecessor graph, *ii)* identifier of graph (subgraph) and node in the main graph." Subgraph frequency is determined by count of different graph in its Lists. Lists are quick responsive but when it comes to scalability EL are not a good choice. Here comes concept of Recomputed Embeddings i.e. RE which keeps an 'active graph's set' in which occurrences are computed repeatedly.

4. FSM Algorithms: Survey

FSM problem is being studied since later 1990s, since then this problem has been addressed in many ways and directions with many approaches. Among those approaches most popular ones are Apriori-based and pattern growth based approaches. Also algorithms differ in many ways like the input graph type, search strategy used by them, frequency counting method, or graph representation method. Hence many algorithms exist using different approaches. Every algorithm has got some limitations and performs better in specific application scenario and test cases.

5. Algorithmic Approach Based Classification

FSM algorithms are popularly classified into two categories:

5.1. Apriori based approach

The basic idea behind Apriori–based FSM Algorithms is to join such two subgraphs which are frequently occurring e.g. G1 and G2 and check for the resultant graph (whose size is exactly one vertex more than other two frequent subgraphs G1and G2) is frequent or not. Apriori based algorithms are recursive in nature because they need subgraph with 'k-1' size to determine subgraph with size 'k'. Such algorithms many times result in generation of multiple candidates.

5.2. Pattern Based Growth Approach

To prevent the overhead caused by merging two subgraphs in Apriori approach based algorithms, Pattern Growth Algorithms were developed. "The pattern Growth FSM algorithm extends frequent graph by adding new edge, in each possible position." Since the pattern-growth approaches uses edge-extension technique and a possible problem with this approach is that same subgraph can be discovered many times. This problem was later overcome by using rightmost extension only in which approach extensions take place at rightmost path only.

6. Search Strategy Based Classification

There are two very popular search strategies applied to find out FSGs and do not require any explanation:

- ***A.*** *BFS (Breadth First Search) or Level wise search strategy*
- ***B.*** *DFS (Depth First Search) strategy*

7. Nature of Input Based Classification

The algorithms are classified in two types based on the correctness of the input. One algorithm takes in exact graph sets as input, whereas the second type of algorithms take an uncertain set of graphs as input. Another possibility of classification is based on type of input graph. One type of algorithm takes input as a single large graph, whereas the second type of algorithms takes set of small graphs as input. The third possible correctness of the graph data where it is exact or uncertain.

8. Totality of Output Based Classification

Based on the set of the FSGs discovered, the algorithms are of two types. The first variety returns the complete set of FSGs, whereas another variety of solution returns a partial set of FSGs.

9. Existing FSM Algorithms

WARMR is probably first algorithm for FSM proposed in 1998 was an Apriori based solution uses ILP and level wise i.e. BFS and generates possible candidates based on subgraph size however this solution missed many FSGs. [1]

AGM is an Apriori approached solution developed in 2000, makes use of BFS strategy. So, this algorithm generates candidate graph based on vertex which increases substructure or subgraph size. To serve the requirement of graph representation, AGM uses adjacency matrix concept. Experiment was performed on large chemical compound dataset. AGM accurately mined FSG from input dataset however its complexity was high because of multiple candidate generation. [2]

FARMER algorithm coined in 2001 is a Breadth-First Apriori algorithm which uses trie data structure for graph representation. This approach finds a subgraph and calculate greater possible subgraphs by adding an adjacent edge of child in every possible way. Farmer performs much better than WARMR, the previous mining algorithm. [3]

MoFa FSM Algorithm developed in 2002 is a Lex-DFS along with structural pruning based frequency counting & Pattern Growth based solution which takes Set of Graphs as inputs and produces All Frequent Subgraphs (Now onwards referred as FSG) as output. It uses Adjacency list representation of Graph. This solution worked well on large chemical atomic structures. One restraint in this algorithm was that its generated frequent graphs aren't exactly frequent. [4]

CloseGraph FSM Algorithm proposed in 2003 is a technical advancement of **gSpan** [5] algorithm & takes Set of Graphs as inputs and produces Frequent Graphs which are closed and connected in nature as output. Frequency counting is done in Lex-DFS way & uses Adjacency list for graph representation. It uses concept of subgraph enumeration for candidate generation. Time overhead for failure detection is one of its notable drawback. [6]

Kuramochi et al. [7] in 2004 tossed concept of heuristic FSM algorithm termed as **GREW** to escalate limitations of contemporary methods. GREW operates on bulky graph, finds patterns for every attached subgraph which has vertex-dis-joint embedding in enormous number. Their results proved GREW to be efficient and scalable as well. An extended version of this approach was further surfaced in 2006 as **Dynamic GREW** [8] which extends its working on dynamic graphs.

FSG is Apriori based FSM algorithm in which edges are considered as frequent item-set. Thus size of graph can be increased only by adding solo edge to the subgraph and new candidate graph will always be greater in size than preceding ones. FSG uses sparse representation of graph for storing candidate graphs and frequent subgraphs. Adjacency list representation is used for storing input graph. Canonical label technique is used to verify whether two graphs are isomorphic graphs or not. Isomorphic testing performed in FSG is very costly and multiple candidates are also generated during candidate generation process. [9]

In year 2005 **SUBDUE** mining algorithm came into picture. SUBDUE uses Beam Search & takes Single large graph as inputs and produces complete set of FSG as output. Frequency counting is done by computing code-length with least description (MDL). It used Adjacency matrix for graph representation. This algorithm can mine very less count of pattern which proves to be its restriction. [10]

Proposed in 2005 and published in 2007, **gFSG** was computationally better approach developed specifically for geometric patterns (subgraphs). This heuristic approach performed in time effective way still neglected few geometric patterns. [11]

mSpan by Y. Li et al. in 2009 takes Set-of-Graphs as inputs and produces FSGs as output. Frequency counting is based on Lex-DFS approach & Adjacency list are used for graph representation. It makes use of FP-growth idea however major drawback was found to be that it's working is limited to labelled graphs only. [12]

Another solution floated same year with **JPMiner** by Yong Liu et al. in 2009 which gets set-of-Graphs as inputs and yields frequent jump patterns as output. Similar to mSpan in JPMiner also Frequency counting is based on Lex-DFS approach & Adjacency list are used for graph representation. Input dataset was a normal graph dataset. This algorithm sometimes generates much smaller set of jump pattern which result in increase of time complexity of algorithm. [13]

"Temporal Subgraph Patterns" (TSP-algorithm) a novel approach by Hsieh et al. in 2010 uses "Heterogeneous Information Networks" (**HIN)** where input is dynamic Set-of-Graphs and Closed Temporal FSG are output. Frequency counting is done by Temporal Subgraph Patterns tree. It uses Adjacency list for graph representation. This algorithm takes extra overhead to check closed temporal patterns which result in increase of time complexity of algorithm. [14]

RP-FP & RP-GD FSM Algorithm developed by Yong Liu et al. in 2011. Both of these algorithms takes in Static Set-of-Graphs and gives Representative Graphs and Jump Pattern as output. Frequency counting is done in popular Lex-DFS order & Adjacency list is used for graph representation. This approach makes use of previously discussed CloseGraph algorithm for FSG mining. These algorithm takes overhead time to summarize the pattern. [15]

MultiObjectiveGBDM in 2014 proved to be a better approach as it used MultiObjectiveGraphDataMining instead of conventional UniObject model and used unary as well as binary metrics for mining task, it implemented previous mining algorithms in multi-objective manner and result were better in each case. [16]

DGP "**D**ensity-based **G**raph **P**artitioning strategy" in 2015 was scalable approach able to mine FSGs of larger graphs with help of MapReduce framework, but in this approach there were few FSG miss compared to previous sequential algorithms but performance was improved considerably. [17]

BigDataFSM by Aridhi and Nguifo in 2016 is also Hadoop based comparison which integrates with **PARMA** [18] a parallel

frequent item-set mining method & **PEGASUS** [19] which is open-source graph-mining library. [20]

IncGM+ FSM Algorithm is developed by Ehab Abdelhamid et al. in 2017 is a remarkable algorithm as it works effectively with evolving graphs. This takes Set of Graphs as inputs and produces FSG as output. This algorithm maintains MNI Table for Pattern Matching [Figure 2]. Conventional and popular Adjacency lists are used for graph representation. Input dataset was a real directed graphs. This algorithm takes memory overhead to maintain maximal frequent subgraphs (MFS) and Minimal infrequent subgraphs (MIFS). [21]

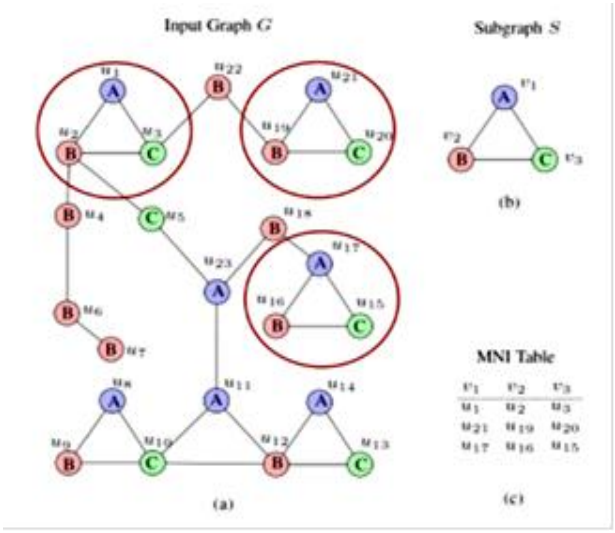


Figure 2: MNI Table Maintenance in IncGM+

Ap-FSM proposed in 2018 gives us a parallel solution for problem of FSM. It uses **Pregel** framework based on "Bulk Synchronous Parallel (BSP) algorithm" [22] & Apache Giraph. It processes single large graph, and does subgraph pruning to avoid unwanted communication giving it high scalability and huge data processing though load balancing needs somehow improvement. [23]

MuGraM by Vijay et al. takes FSM to a level where multigraphs can be processed for subgraph mining and it tested many datasets like amazon, citeseer etc. this backtracking based algorithm keeps support computation as least as possible and subgraph pruning reduces search space complexity making it better than few other approaches. [24]

10. Application Areas of FSM

FSM Algorithms are used in several areas like, Web analysis, Chem-Informatics, Wireless Networks, Telecommunication Networks, Financial Network Analysis, Social behavior Analysis, sentiment analysis, protein structure analysis etc. Figure 3 demonstrates various domains in which FSM algorithms are applicable and which Algorithm is suitable for that domain.

Social behavioral analysis with FSM has big present and future scope.

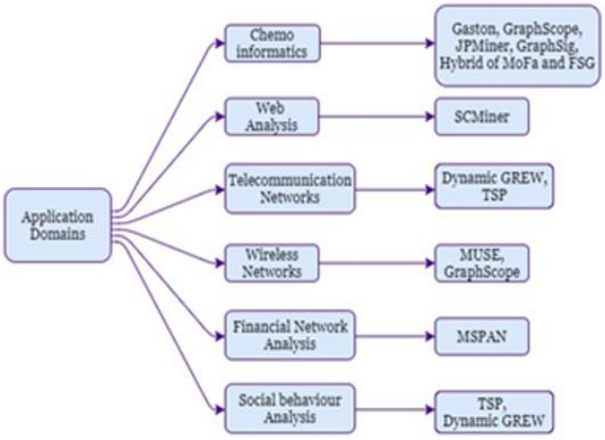


Figure 3: Application Areas of FSM

11. Conclusion

In this Paper, we studied several FSM algorithms, their merits, demerits, behavior and application. FSM is not only part of Computer Science it has its application in real world problems and in many inter-disciplinary domains e.g. cheminformatics, Economics case studies, bioinformatics etc. Lot of work has been done and there is still possibility of ample amount of advancement in present FSM algorithms. There exist less number of research work which are able to work on dynamic or evolving graphs, similarly, parallel implementation of these algorithm will also save lot of our time. Also currently available algorithms just do better from their predecessor but none of them focus on the problem of NP-completeness of FSM.

References

[1] L. Dehaspe, H. Toivonen, and R. D. King. Finding Frequent Substructures in Chemical Compounds. Pages 30-36. AAAI Press, 1998.
[2] A. Inokuchi, T. Washio, and H. Motoda, "An apriori-based algorithm for mining frequent substructures from graph data," Proc. Fourth European Conf. Principles of Data Mining and Knowledge Discovery (PKDD), pp. 13-23, 2000.
[3] S. Nijssen and J. N. Kok, "Faster association rules for multiple relations," IJCAI, pp. 891–896, 2001.
[4] Borgelt and M. R. Berhold, "Mining molecular fragments: finding relevant substructures of molecules," Proc. 2nd IEEE Int'l Conf. Data Mining (ICDM '02), pp. 51-58, 2002.
[5] X. Yan, J. Han, "gSpan: graph-based substructure pattern mining," Proc. 2nd IEEE Int'1 Conf, 2002.
[6] X. Yan, J. Han, "CloseGraph: mining closed frequent graph patterns," Proc. Ninth ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD), pp. 286-295, 2003.
[7] M. Kuramochi and G. Karypis,"GREW—a scalable frequent subgraph discovery algorithm," 2004.
[8] K. M. Borgwardt, H. P. Kriegel, P. Wackersreuther,"Pattern mining in frequent dynamic subgraphs," ICDM 2006.
[9] M. Kuramochi and G. Karypis, "Frequent Subgraph Discovery," Proc. IEEE Int'l Conf. Data Mining (ICDM), pp. 313-320, 2004.
[10] N. S. Ketkar, L. B. Holder, D. J. Cook,"Subdue: compressionbased frequent pattern discovery in graph data," 2005.
[11] M. Kuramochi, G. Karypis, "Discovering frequent geometric subgraphs", Information Systems, Volume 32, Issue 8, 2007, Pages 1101-1120
[12] Y. Li, Q. Lin, G. Zhong, D. Duan, Y. Jin and W. Bi, "A Directed Labeled Graph Frequent Pattern Mining Algorithm Based on Minimum Code," 2009 Third International Conference on Multimedia and Ubiquitous Engineering, Qingdao, 2009, pp. 353-359.
[13] Y. Liu, J. Li, H. Gao, "JPMiner: Mining Frequent Jump Patterns From Graph Databases," In the proceedings of Sixth International Conference on Fuzzy Systems and Knowledge Discovery 2009.

[14] H. Hsieh,C. Li, "Mining Temporal Subgraph Patterns in Heterogeneous Information Networks," IEEE International Conference on Social Computing / IEEE International Conference on Privacy, Security, Risk and Trust, 2010.
[15] J. Li, Y. Liu, and H. Gao, "Efficient Algorithms for Summarizing Graph Patterns," IEEE Transactions On Knowledge And Data Engineering, Vol. 23, No. 9, September 2011.
[16] P. Shelokar, A. Quirin, Ó. Cordón, Three-objective subgraph mining using multiobjective evolutionary programming, Journal of Computer and System Sciences, Volume 80, Issue 1, 2014, Pages 16-26.
[17] S. Aridhi, L. d'Orazio, M. Maddouri, E. M. Nguifo, "Density-based data partitioning strategy to approximate large-scale subgraph mining," Information Systems, Volume 48, 2015, Pages 213-223.
[18] M. Riondato, J.A. DeBrabant, R. Fonseca, E. Upfal, PARMA: a parallel randomized algorithm for approximate association rules mining in MapReduce, in: Proceedings of the 21st ACM International Conference on Information and Knowledge Management, CIKM'12, ACM, New York, NY, USA, 2012, pp. 85–94.
[19] U. Kang, C.E. Tsourakakis, C. Faloutsos, PEGASUS: a peta-scale graph mining system implementation and observations, in: Proceedings of the 2009 Ninth IEEE International Conference on Data Mining, ICDM'09, IEEE Computer Society, Washington, DC, USA, 2009, pp. 229–238.
[20] S. Aridhi, E. M. Nguifo, "Big Graph Mining: Frameworks and Techniques," In Big Data Research, Volume 6, 2016, Pages 1-10, ISSN 2214-5796.
[21] E. Abdelhamid, M. Canim, M. Sadoghi, B. Bhattacharjee, Y. C. Chang and P. Kalnis, "Incremental Frequent Subgraph Mining on Large Evolving Graphs," in IEEE Transactions on Knowledge and Data Engineering, vol. 29, no. 12, pp. 2710-2723, Dec. 1 2017.
[22] G. Malewicz , M. H. Austern , A. J. C. Bik , J. C. Dehnert , I. Horn, N. Leiser, "Pregel : A system for large-scale graph processing," In The pro-ceedings of the ACM SIGMOD international conference on management of data (pp. 135–145), 2010.
[23] V. Bhatia, R. Rani, Ap-FSM: A parallel algorithm for approximate frequent subgraph mining using Pregel, Expert Systems with Applications, Volume 106, Pages 217-232. 2018
[24] V. Ingalalli, D. Ienco, P. Poncelet, Mining frequent subgraphs in multigraphs, Information Sciences, Volumes 451–452, Pages 50-66, 2018.