International Journal of Engineering, Science and Mathematics
Vol. 6 Issue 8, December 2017 (Special Issue)
ISSN: 2320-0294 Impact Factor: 6.765
Journal Homepage: http://www.ijesm.co.in, Email: ijesmj@gmail.com Double-Blind
Peer Reviewed Refereed Open Access International Journal - Included in the International Serial Directories Indexed & Listed at: Ulrich's Periodicals Directory ©, U.S.A., Open J-Gage as well as in Cabell's Directories of Publishing Opportunities, U.S.A

Mining Frequent Patterns in Web log data Using Gaston Algorithm

N. Jaya Lakshmi*
Dr. P. Padmaja**
Dr. G. Jaya Suma***

Abstract

Web based applications are growing along with technology and an enormous amount of data involved therein led to the development of methods to discover interesting frequent patterns based on user's needs. Web mining is an application of data mining techniques to the World Wide Web. Discovering of interesting frequent usage patterns from web log data leads to better understanding of user behavior and aids to serve the users in a better way. This process of discovering frequent usage patterns is constructed in three phases like preprocessing, pattern discovery using data mining algorithm and pattern analysis. In this paper, we discussed some of data mining algorithms to discover frequent patters from web log data and GASTON algorithm is applied to extract frequent patterns from the web log data. For the implementation, web log files are extracted from server and then different graphs are generated from that data.

Keywords:

Web mining,
frequent patterns,
graph mining,
GASTON algorithm.

Author correspondence:

N. Jaya Lakshmi
Associate Professor,
Department of CSE,
VITAM College of Engineering, Visakhapatnam

1. Introduction

Data mining aims to discover interesting or useful patterns which are hidden in a given dataset. Graph mining is one of the sub area of data mining in which data mining concepts and techniques can be widely used. Graphs are more suitable to represent complex data set like chemical compounds, social networks, WWW and XML documents. Frequent sub graph mining (FSM) which is most dominant concept in Graph mining. Frequent pattern discovery in web log data is to obtain useful information about the navigational behavior of the users. The obtained information can be used for advertising purposes, for creating dynamic user profiles, site modifications, business intelligence, system improvement and personalization.

Weblog is unstructured data and therefore mining relevant information from weblog is a very challenging task. Web contains billions of web pages. The web access behavior of one user is different from that of another. By analyzing the users' data, the web pages can be personalized according to individual web users' interest. Personalizing the web page gives various advantages in this fast era such as low search time, less data transfer, higher availability of data, lower bandwidth traffic, targeted advertisement and identifying the threaded web users and high web user's satisfaction.

*Associate Professor, Department of IT, GITAM University
**Professor&HOD, Department of IT, JNTU Kakinada

A Graph G consists a set of nodes V and Edges E, denoted as G ={ V_G, E_G }, defined as V= {v | v ЄV}, E={(v_1,v_2)|v_1,v_2Є V and $v_1 \neq v_2$}, where any edge (v_1,v_2) is equal to (v_2,v_1) since all of the edges in G does not have any direction. The adjacency matrix A of G is defied as

$$[A]_{ij} = \begin{cases} 1 & \text{if } (v_i, v_j) \in V \\ 0 & \text{Otherwise} \end{cases}$$

Where v_i and v_j are nodes in G.

A graph $G^1 = (V^1, E^1)$ is said to be a sub graph of G = (V,E) if $V^1 \subseteq V$ and $E^1 \subseteq E$.

Graph Isomorphism: A Graph $G^1 = (V^1, E^1)$ is said to be isomorphic to another graph G = (V,E) if there exists a bijective function $\varphi : V^1 \rightarrow V$, i.e, both injective (into) and surjective (onto), such that

$(u,v) \in E^1 \leftrightarrow (\varphi(u), \varphi(v)) \in E$

For all $u \in V^1$, $L(U) = L(\varphi(u))$

For all $(u,v) \in E^1$, $L(u,v) = L(\varphi(u), \varphi(v))$

If the function φ is only into, but not onto, then mapping φ is a sub graph isomorphism from G^1 to G. i.e., G^1 is sub graph isomorphic to G, denoted as $G^1 \subseteq G$

A standard approach for handling graph isomorphism is to map each graph into a unique string representation, which is known as *canonical labeling*. That is, if two graphs are isomorphic then their canonical code must be the same.

Given a graph data set, L={ G_0, G_1, G_2,…. $G_{0=n}$}, support(g) denotes the number of graphs in L in which g is a sub graph. The problem of frequent sub graph mining is to find any sub graph g such that *support(g)>=minsup*, where minsup is a user specified threshold.

There are two main challenges in FSM. First is to generate candidate sub graphs systematically. If we consider sub graphs with m vertices, then there are $O(m^2)$ possible edges. The number of possible sub graphs with m nodes is $O(2^{m2})$ and many of these sub graphs may be isomorphic tom each other and the number of distinct sub graphs may be less. So, here we need to have a mechanism for graph isomorphism, to perform non redundant sub graph enumeration. This mechanism of graph isomorphism will enable us to remove the duplicate graphs. The second one is to count the support of a graph in the data base. This process involves sub graph isomorphism checking.

Considering a single minimum support threshold, *rare item problem* occurs, which means that useful sub graphs with low supports cannot be extracted when a minimum support threshold is high. But, every element in the real world has an importance, even though its frequency (support) is low. And various elements of graph data base may need to have different thresholds based on their characteristics. FSM techniques can be divided into two categories:

(i) Apriori approach

(ii) Pattern Growth approach

An apriori approach for mining frequent sub graphs consists the following steps.

Candidate generation: In this process, candidate k-sub graph is obtained by merging pairs of frequent (k-1) sub graphs.

Candidate pruning: In this step, all candidate k-sub graphs, which contain infrequent (k-1) sub graphs, are discarded.

Support counting: This the process of counting the number of graphs in database that contain each candidate.

Candidate elimination: This step discards all the candidate sub graphs, whose frequency (support) is less than the user specified threshold.

According to *Anti-monotonicity* property, if a graph is frequent, then all of its sub graphs must be frequent. So, all frequent (k-1) candidates (graphs) are used to generate candidate k-sub graph. Candidates can be generated by vertex growing or edge growing. Vertex growing is the process of generating a new candidate by adding a new vertex in to an existing frequent sub graph. Thus, when the candidates are generated by vertex growing, there is substantial increase in the number of candidates sub graphs. Edge growing inserts a new edge to an existing frequent sub graph. When this method is used, the number of candidate sub graphs tends to be smaller than those produced by the vertex growing method.

When there is a number of candidate sub graphs, out of some of may be redundant. So, there is a necessity for reducing the generation duplicate sub graphs. This involves the process of Graph Isomorphism, to check whether two graphs are matched. If two or more graphs are matched, then only one of them is used for the next level and rest of them are not considered. Thus, key point in this candidate generation is to generate candidates systematically without redundancy. That is, each graph should be generated at most once.

2. Literature Survey

The primary goal of data mining is to extract hidden, but useful, knowledge from data. The data to which data mining may be applied can be categorized according to its representation mechanism: vectors, tables, texts, images, and so on. Data can also be categorized as being structured (e.g. molecule data), semi-structured (e.g. XML document collections) and non-structured (e.g. sound or video). Graph representation is suitable for structured data.

Because of the ease with which structural data can be represented using a graph format, substantial research work has been directed towards the mining of graph data, also referred to as graph based data mining or graph mining.

Frequent sub graph mining is one of the most common topics of graph mining. Frequent sub graph mining aims to identify all sub graph patterns whose occurrences within a graph data set are above some user defined threshold. These sub graph patterns are called frequent sub graphs. The number of occurrences, also called the frequency, of each sub graph pattern is computed by a support measure. Theoretically, frequent sub graph mining can be formulated as a search in a search space, modelled by a lattice, consisting of all possible sub graph patterns. Because the number of possible frequent sub graphs increases exponentially with the size of the graph, completely traversing the search space is computational intractable. Most frequent sub graph mining algorithms thus adopt a user specified support threshold to prune this combinatorial search space, i.e. the support metric is used to separate infrequent sub graphs from the frequent ones.

Frequent sub graph mining plays an essential role in many graph mining applications such as chemical compound analysis [Huan et al., 2004c, Deshpande et al., 2005], document image clustering [Barbu et al., 2005], software bug isolation [Liu et al., 2005, Eichinger et al., 2008], web content mining [Schenker et al., 2004], social network mining [Mukherjee and Holder, 2004, Yang et al., 2006, Lahiri and Berger-Wolf, 2007], email mining [Aery and Chakravarthy, 2005a,b], and anomaly detection [Noble and Cook, 2003, Eberle and Holder, 2007].

Frequent sub graph mining uses the support metric. If the Support threshold is low, then large number of patterns is often generated. Further, there is a redundancy on the resulting collection of patterns. Furthermore, analyzing this large collection of patterns is difficult, time consuming and resource intensive. A high support threshold will reduce the number of patterns detected, but at the risk of missing significant patterns.

In the weighted frequent sub graph mining some vertexes and/or edges can be considered to be much more significant than others; consequently, by using weightings, the search space can be reduced in such a way that the most interesting patterns are still discovered. By integrating weight constraints into the work of frequent sub graph mining, we may expect that a smaller set of weighted frequent sub graphs can be discovered within a reasonable time-limit. This can be achieved with the established frequent sub graph mining algorithms.

gSpan

gSpan [Yan and Han, 2002] used a canonical representation, to uniquely represent each sub graph. The algorithm used DFS lexicographic ordering to construct a tree-like lattice over all possible patterns, resulting in a hierarchical search space called a DFS code tree. Each node of this search tree represented a DFS code. The (k + 1)-th level of the tree had nodes which contain DFS codes for k-sub graphs. The k-sub graphs were generated by one edge extension from the k-th level of the tree. This search tree was traversed in a DFS manner and all sub graphs with non-minimal DFS codes were pruned so that redundant candidates are avoided. Instead of keeping embedding lists for each discovered sub graph, gSpan only preserved the transaction list for each discovered pattern and sub graph isomorphism detection was only applied to graphs within the list. Thus, gSpan saved on memory usage because keeping embedding lists required a lot of memory resources to store all embeddings of a sub graph in the database. gSpan was adopted as the “base" algorithm into which many of the proposed weighted FSM techniques described in this thesis were incorporated for evaluation purposes.

GASTON

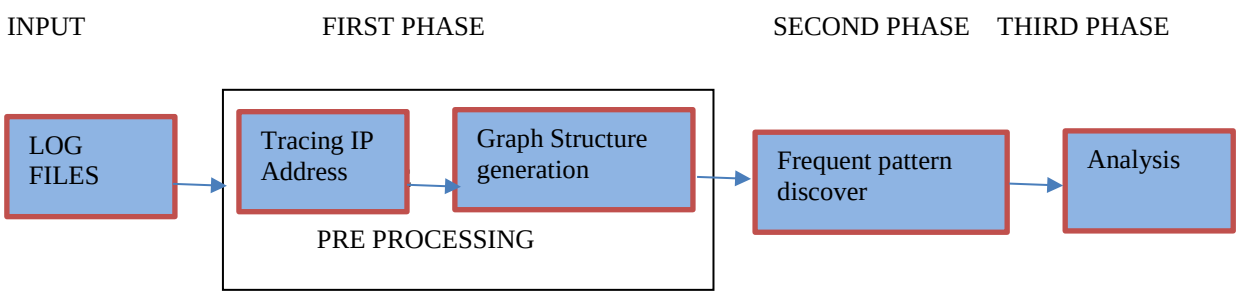
Nijssen and Kok [2004] observed that most frequent sub-structures in molecular databases are free trees. Based on this observation, they introduced GASTON, which integrated frequent path tree and graph mining into one algorithm. The algorithm divided the FSM process into three phases: path mining, then sub tree mining, and finally sub graph mining. Thus, GASTON operated best when the graphs are mainly paths or trees, because the more expensive sub graph isomorphism testing is only encountered in the sub graph mining phase. GASTON used a hash function and graph isomorphism detection to identify the duplicates. GASTON also recorded the embedding lists so as to only grow patterns that actually occurred in the transaction graphs, thus saving on unnecessary isomorphism testing.

3. Research Method

The objective is to find all the frequent patterns from this web log data. We can get the required data from a web server. Access log files at server side records the browsing history of site visitors. These log files gives the basic information, which can be used for mining purpose.

The overall process of finding frequent patterns in a web data, has been done in three phases namely Preprocessing, Extracting and Analysis

First, in the preprocessing phase, the web log data has been converted in to web transaction data base by tracing the IP address. Then, the graph structure constructed by using web transaction data base. In this paper, the weight of edges of the graph are considered same, even though weight of edges may be the frequency and cumulative duration of web page. Second, in the Mining phase, GASTON algorithm will be applied to find frequent patterns by inputting the generated web log graphs. Third, in the Analysis, the frequent patterns mined from the web log data will be analyzed using time and the number of patterns mined for different thresholds. The following fig (i) shows the overall process of discovering frequent patterns in the web log data.



Fig(i) Overall process of finding frequent patterns of web log data

We have taken a data base of web access logs of a website www.gleezetech.com from January 2014 to December 2014

3.1 Preprocessing Phase: Access log files gives the information about different user's navigation history. This information gives the browsing behavior of site visitors.

Normally, web access log file contains the information about IP address, Access time, HTTP request method, Path of source on the web server (URL), Protocol used for transmission, Status code, Number of bytes transmitted

Irrelevant information which is useless for mining process can be removed from the log files. For example, if the status code is 400 (bad request), 403(forbidden), 404(not found), 503(out of resources), 100 (continue), 200(request accepted) etc. Here, we have considered only the status code 200, means which requests are accepted. Remaining status codes are deleted. We focused only on four attributes IP address, access time, URL and only the status codes 200

After cleaning and obtaining the relevant data, this will be converted in to web user transaction data base by tracing the IP address. We have used MS-Excel, by importing the web log data which is in text form, and converted into web transaction data base for each IP address.

The graph structure will be generated, as every node is the unique web page and the edges are generated based on the page sequence viewed by the user. We considered equal weight for each edge of a graph. Here, based on 4 days transaction database, 4 graphs are generated manually. For each day, 5 random users, based on IP address and their navigation, graph is generated. These 4 graphs are given as input to the next phase. Fig (ii) shows this preprocessing phase of the overall process.

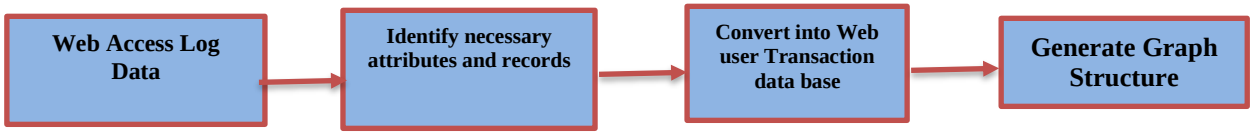


Fig (ii) Preprocessing Phase in discovering frequent patterns

3.2 Extracting Phase: This is the actual mining stage. Here, we will apply the GASTON mining algorithm to find frequent pattern from the web log graphs. Here, the input is the following 4 graphs from the web log data. Output from this phase is a set of frequent patters on a given threshold. These patterns may be paths, free trees and cyclic graphs. Fig(iii) shows mining phase of the overall process.

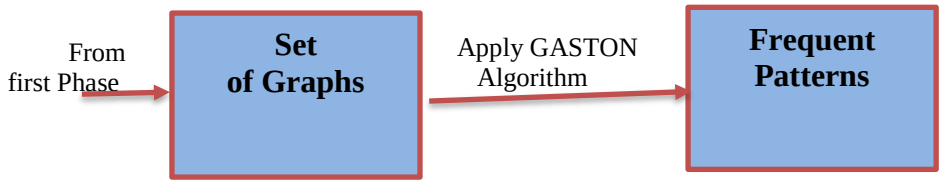


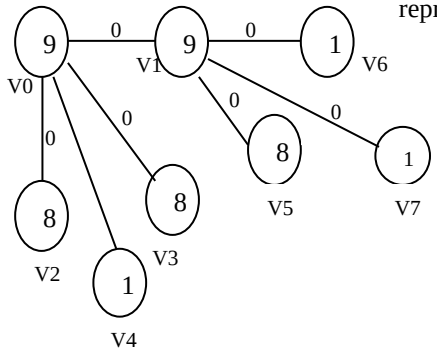
Fig (iii) Extracting Phase – Actual mining phase

3.3 Analysis Phase: In this stage, the obtained frequent patterns from the web log data will be analyzed using time, number of structures & may compare with other algorithms. From this obtained user access patterns, we can identify the typical browsing behavior of the users and we can make prediction about desired pages.

4. Results and Analysis

To execute GASTON algorithm to obtain frequent patterns, the input of the graph database is given in the following format:

Example: G1=(V1,E1) V1={v0,v1…v7}, this graph G1 can be represented as follows:



G1

V 0 9
V 1 9
V 2 8
V 3 8
V 4 1
V 5 8
V 6 8
V 7 1

e 0 1 0
e 0 2 0
e 0 3 0
e 0 4 0
e 1 5 0
e 1 6 0
e 1 7 0

V 1 9 indicates vertex one with label 9

e 1 7 0 indicates there is an edge from vertex 1 to vertex 7 with an edge label 0

GASTON Algorithm is executed for different thresholds. For the given 4 graphs, when the threshold is 2, there are 17 frequent URLs, 100 paths, 24 free trees and 13 cyclic graphs. When the threshold is 3, we have obtained 14 frequent URLs, 37 paths, 5 free trees and 2 cyclic graphs.

Frequent URLs	Threshold	Paths	Free trees	Cyclic graphs	Execution time
17	2	100	24	13	0.017473s
14	3	37	5	2	0.0011s

5. Conclusion

In this paper, GASTON algorithm is applied to discover frequent patterns in web log data. From this paths, free trees and cyclic graphs are generated in sequence. The popularity of this algorithm is due to its time complexity and moderate than other FSM algorithms. This application can be extended to extract the interesting patterns from the frequent patterns. New parameters can be added to generate the interesting patterns along with GASTON algorithm.

References

[1] D. J. Cook and L. B. Holder, "Substructure discovery using minimum description length and background knowledge" Journal of Artificial intelligence Research, 1, 1994, 231-255.

[2] A. Inokuchi, T.Washio, and H. Motoda. An apriori-based algorithm for mining frequent substructures from graph data. In PKDD'00.

[3] M.Kuramochi and G. Karypis .Frequent Sub graph Discovery. In ICDM'01.

[4] J. Huan, W.Wang, and J. Prins. Efficient mining of frequent sub graph in the presence of isomorphism. UNC computer science technique report TR03-021, 2003. Computer Science & Information Technology (CS & IT) 201

[5] J. Huan, W. Wang, J. Prins, and J. Yang. Spin: Mining maximal frequent sub graphs from graph databases. UNC Technical Report TR04-018, 2004.

[6] M. Kuramochi and G. Karypis. GREW A Scalable frequent sub graph discovery algorithm. Technical Report 04-024, University of Minnesota, Department of Computer Science, 2004.

[7] C. 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.

[8] X. Yan and J. Han.gSpan: Graph-based substructure pattern mining. Proc. 2nd IEEE Int'l Conf. Data Mining (ICDM '02), pp. 721-724, 2002.

[9] L. T. Thomas, S. R. Valluri, and K. Karlapalem. Margin: Maximal frequent sub graph mining. Proc. 6th IEEE Int'l Conf. Data mining (ICDM '06), pp. 1097-1101, 2006.

[10] M. Kuramochi and G. Karypis.Grew- A scalable frequent sub graph discovery algorithm. In ICDM, pages 439–442,2004.

[11] Thomas, L., Valluri, S. and Karlapalem, K., Isg: Item set based sub graph mining. Technical Report, IIIT, Hyderabad, December2009.

[12] Kuramochi, M. and Karypis, G., Finding frequent patterns in a large sparse graph. Data Min. Knowledge Discovery, 2005, (3),243–271.

[13] ZhaonianZou, Jianzhong Li, Hong Gao, and ShuoZhang : Frequent Sub graph Patterns from Uncertain Graph Data. IEEE Transactions On Knowledge And Data Engineering, Vol. 22, No. 9, September 2010.

[14] Nijssen, S. and Kok, J., Faster association rules for multiple relations. In IJCAI'01: Seventeenth International Joint Conference on Artificial Intelligence, 2001, vol. 2, pp. 891–896.

[15] Nijssen, S. and Kok, J., A quick start in frequent structure mining can make a difference. In Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2004, pp. 647–652.

[16] Cordella, L.P., Foggia, P., Sansone, C. and Vento, M. 2001. An Improved Algorithm for Matching Large Graphs, In Proceedings of the 3rd IAPR-TC15 Workshop on Graph-based Representation in Pattern Recognition, 149–159.

[17] Chuntao Jiang, FransCoenen and Michele Zito, A Survey of Frequent Sub graph Mining Algorithms :The Knowledge Engineering Review, Vol. 00:0, 1–31.c 2004.

[18] Varun Krishna, N. N. R. RangaSuri and G. Athithan,A comparative survey of algorithms for frequent sub graph discovery, Current Science, Vol. 100, No. 2, 25 January 2011

[19] Yuhua Li Quan Lin Gang ZhongDongshengDuanYanan Jin Wei Bi, A Directed Labeled Graph Frequent Pattern Mining Algorithm based on Minimum Code. In the proceedings of Third International Conference on Multimedia and Ubiquitous Engineering 2009.

[20] Jianzhong Li, Yong Liu, and Hong Gao, Efficient Algorithms for Summarizing Graph Patterns: IEEE Transactions On Knowledge And Data Engineering, Vol. 23, No. 9, September 2011.