Bon iterative graph feature mining for graph indexing

A. Pankaj Moses Monickaraj[1], K. Vivekanandan[2], D. Ramya Chitra[3]

[1]Doctoral Scholar,Department of Computer Science, Bharathiar University, India
[2]Professor, BSMED, Bharathiar University, India
[3]Assistant Professor, Department of Computer Science, Bharathiar University, India
pankajmoses@hotmail.com

Abstract. Extracting Sub graph from a colossal graph database is one of the key problems in Graph Mining. Feature mining approaches (i.e) mine at once algorithms like G-index, FG index, gSpan build index for any graph data. But, if incase of changes in the data (updates reach or change in memory size), the graphs have to be upgraded as well. Quiet obviously the index has to be updated. It is quite expensive to construct and deploy a new Graph Index from scratch instead iterative mining algorithms can be used. As the sizes of these algorithms are small, they can be iteratively used then and while at consistent intervals. These provide the updated list of features from the graph database. Iterative graph feature mining algorithm [1] is one of the key algorithms playing dominant role in feature extraction. After extracting the updated features [3], they have to be re-indexed and inserted in proper into the index for which various search algorithms can be used. In this paper, an improved iterative graph feature mining is proposed and tested with AIDS and e-molecules datasets which is further compared with the existing results.

Keywords: Graph, Graph Mining, Graph Indexing.

1. Introduction

In the recent days scientific and technological advances have revealed that plenty of certain data or uncertain data or may be patterns can be modeled as graphs. As a result, it is of special interest to process graph containment queries effectively on large graph databases. Consider a graph database G, let q be a query (i.e) the set subgraph relevant to the query q. All the subgraphs relevant to the query q is retrieved in G which contain q as subgraph(s). The isomorphism testing can be done in case if the number of subgraph are vast and any of the indexing mechanism [1] – [5] can be opted to index the complete data in G so that it can be easily accessible. In [6], a frame work is pictured with indexing, iterative mining [7] and re-indexing. In fig [2] in [7], an iterative algorithm was used to extract the updated graph features. In this paper, an improved iterative mining algorithm is proposed and tested with two different datasets. (i.e) AIDS antiviral dataset and e-molecules dataset.

Tendencias Tecnológicas en Computación
Research in Computing Science 66, 2013, pp. 55-62
Paper received 27-05-2013 and accepted 05-08-2013

This paper is organized as follows: we introduce the background and preliminaries in Section II. Then we present the set of related work in section III. In section IV, we first introduce the complete overview of Iterative algorithm. Then in section V, we propose Bon Iterative algorithm to show the improved search space. The algorithm is tested with AIDS and e-molecule datasets in Section VI and the results are discussed. Finally, we conclude this paper in Section VII.

1.1 Preliminaries

Two graphs say $g_1 = (v_1, e_1, l_1)$ and $g = (v_2, e_2, l_2)$ are isomorphic to each other if there is a bijection which is a mapping such that a pair of adjacent vertices u_1, v_1 in g_1 is mapped to a pair of adjacent vertices u_2, v_2 in g_2 where $L(u_1) = L(u_2)$, $L(v_1) = L(v_2)$ and $L(E(u_1, v_1)) = L(E(u_2, v_2))$ and vice versa.

Let G be a graph database, a subgraph g_1 is a frequent subgraph iff it maintain the level greater than the defined parameter level, minimum level □|D|. The number of sub graphs which are obtained for a query from G would be the □|D| for the query. A subgraph search is calculated by adding filtering and verifying paradigm.

Let $f = \{f_1, f_2, ..., f_m\}$ be the set of graph features. Graphs from G can be vectorized and are represented as an m dimensional vector Xg = [x1, x2,…., xm], where xi = 1 if fi ⊂ g and xi = 0. Each vector will be the core features which will be used building index.

The candidate set can be obtained [7] from

$$Cq = \text{finD}(pi) = fi \in \text{maxsubgraph}(q,f) \text{nD}(pi) = D(\text{maxSubgraph}(q, f)) \quad (1)$$

$$\text{Maxsub}(q,f) = \left\{ fi \in \frac{f}{fi} \subset q, \nexists x \in g \text{ s.t. } pi \subset x \subset q \right\} \quad (2)$$

(Maximum Sub graph for a query q)

For set queries say Q,

$$\text{minSup}(f, Q) = \{q \in Q / f \in \text{maxSub}(q, f)\} \quad (3)$$

The time taken for processing for a query q,

$$T \underset{\text{resq}}{\longrightarrow} (q) = T \underset{\text{filter}}{\longrightarrow} (q) + \text{Tverif}(C(q)) \quad (4)$$

The experiment is implemented and tested with AIDS and e-molecule datasets in Section 5.

2. Related work

Depth first search code tree is used by gSpan [8] which extracts the features through their subgraph from a graph database. gSpan also prefers on from the initial right most vertices and proceeds further. Xifeng Yan et.al has proposed gindex [10]

which uses the basic sub structure or sub graph whose parameters are greater than the minimal fixed parameter say |D|. Almost all the features present in the graph database G are taken under initial consideration and further they are extended as the graph with related loops through gIndex [10]. GraphSig [11] initially converts the data into feature vectors and further proceeds on mining the sub feature vectors. Iterative feature mining [7], finds a subgraph feature p with maximum domination the maximum and minimum support can be flexibly altered according to the features perspective present in the concern data. In this paper, the performance of the iterative algorithm is tested with the proposed algorithm and tested with datasets.

3. Bon iterative algorithm for indexing

Consider a graph database G, a stable index is obtained by using any of the mines at once algorithms. Consider there is an up gradation in data. In such case, the instead of completely scanning the dataset just the up gradations can be obtained through Bon Iterative Algorithm.

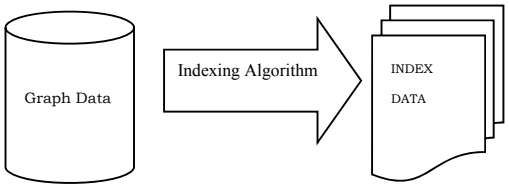

Figure 1. Graph Indexing.

Consider a graph index with feature set pn-1 of size n-1, find a new graph feature p, where p does not belong to pn-1 such that expectation of the verification cost be Tverif. This new features is {p, pn-1} is indexed until it is generated with C(q), where C(q) is the candidate set of the query q. Finally after adding the new feature the set becomes into pn.

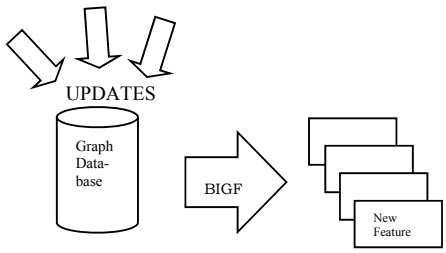

Figure 2. New feature extraction from updated database.

The new feature should be selected as the one with maximizing support through Dijkstras algorithm such that the feature attains the minimal support so as it can be added into the index.

Minimal Support:

Consider a set of query Q and a subgraph feature p0 ∈ P, a graph q0 ∈ Q is a minimal super query of the feature p0 iff the query q0 has a maximal feature of p0. minSup(p0,Q) = {q0 ∈ Q|p0 ∈ maxSub(q0,P)}.by the minimal super queries of p0.

This is how the graph features are extracted to move into index. BIGFMA is tested further with datasets and their experimental results continue in the upcoming section.

3. Experimental results

3.1 Datasets aids antiviral screen dataset

To test the scalability, database index construction etc, we test with the dataset from National Cancer Institute AIDS antiviral, 3 classes (B) July 29, 2004 By Fei Yuan. The potency data is at http://dtp.nci.nih.gov/docs/aids /aids_data.html. NCI A and NCI B have slightly different sets of molecules and different descriptor sets. Otherwise, the potencies are the same.

3.2 E-molecule dataset

To test the scalability, database index construction etc., we have also tested with dataset from chemical page consisting of molecules structure (http://www.e-molecules.com/)

3.3 Screen shots and graphs aids dataset

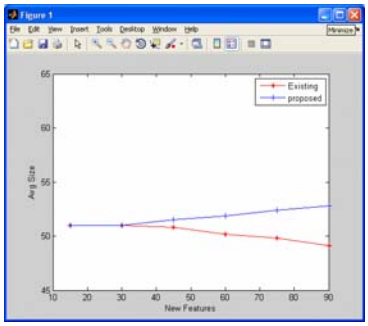

Figure 3. Candidate set size between IGFMA vs BIGFMA.

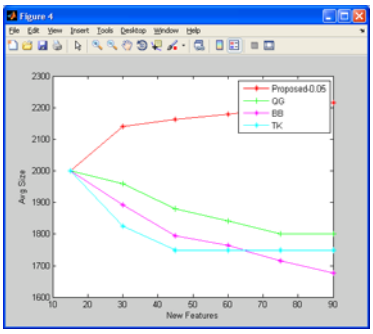

Figure 3. Bon iterative mining at DF(0.05).

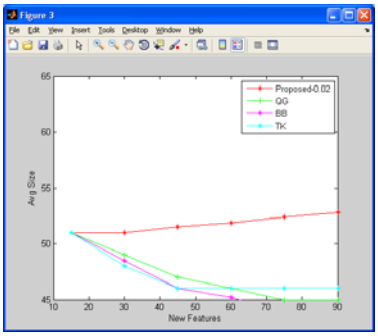

Figure 4. Bon iterative mining at DF(0.02).

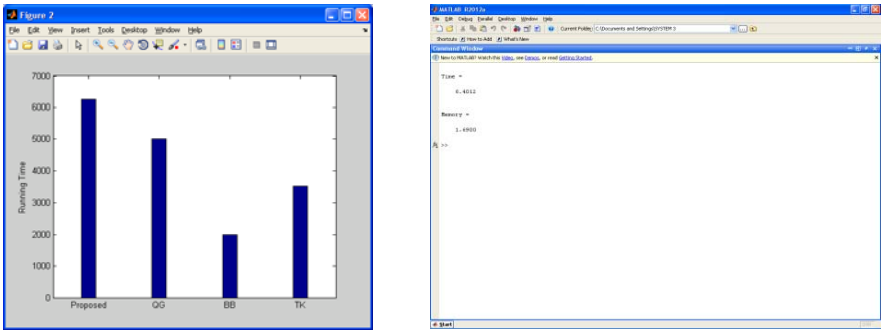

Figure 5. Running time and memory usage of Bon iterative algorithm.

Table 1. Feature count with decreasing MIN-support.

	.1-.8	.8-.6	.6-.4
Proposed	46	103	221
Existing	54	104	247

The counts of the features that are mined are compared with the existing and the proposed algorithm. In case of (.1-.8) 14.8 % of improvement, there is for .8-.6 little improvement and for 6-.4 10.53% of improvement. Hence for this dataset Bon Iterative Algorithm performs better than the existing Iterative algorithm. The time taken for the execution of the algorithm was 6.4012 seconds.

3.4 Molecule dataset

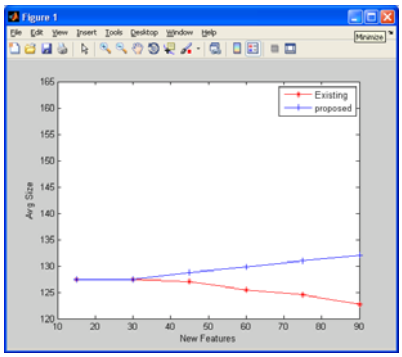

Figure 6. Candidate set Size between IGFMA vs BIGFMA.

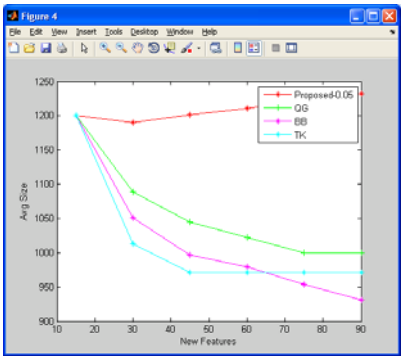

Figure 7. Bon Iterative Mining at DF(0.05).

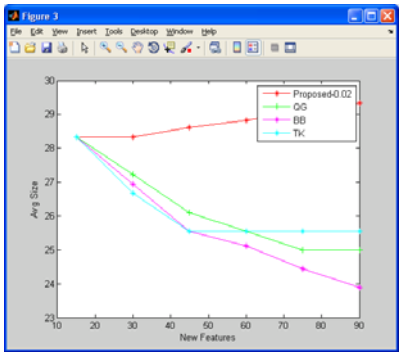

Figure 8. Bon Iterative Mining at DF(0.02).

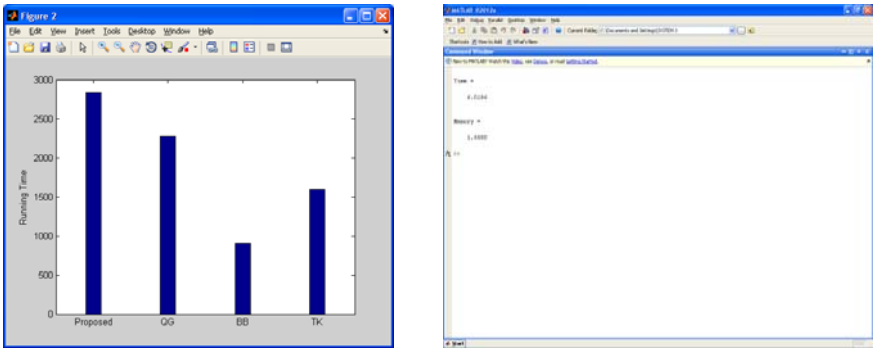

Figure 9. Running time and memory usage of Bon Iterative Algorithm.

Table 2. Feature count with decreasing MIN-support.

	.1-.8	.8-.6	.-.4
Proposed	48	98	220
Existing	54	104	247

In case of e- molecules dataset, the minimal support are found between proposed and existing 11.12 %, 5.7 % and 10.93 % were the improvements found. The time taken for the execution of molecule dataset was 6.0196 seconds.

Environmental requirement: Front end designed with MATLAB (r2012a) with i3 processor and 80 GB memory. RAM size is 4 GB with Intel mother board.

4. Conclusions

Here we have investigated the working of Iterative mining of graph features for the subgraph search problem. From table 1 and from Figure 3,4,5,6 the Proposed Bon Iterative Graph feature Mining Algorithm is providing a better result than the existing Iterative Algorithm when tested with the AIDS dataset. Also, when tested with E-molecules dataset, from table and figure: 7, 8, 9, 10 they provide a better result in time complexity and memory complexity. Hence BIGFMA is better than the Iterative graph feature Mining algorithm for feature extraction and indexing.

Acknowledgement

We would also like to thank Mr. M. Pradeep who helped in programming throughout this work. Would like to thank our friends Mr. Ramkumar, Doctoral Scholar, Ms. Preethi Rajavel, Department of Microbial Bio-Technology, Bharathiar University, Coimbatore- 46, Tamil Nadu, India for their valuable tips during implementation.

References

1. B. Sun, P. Mitra, and C. L. Giles. Irredundant informative subgraph mining for graph search on the web. CIKM, 2009.
2. J. Cheng, Y. Ke, W. Ng, and A. Lu. Fg-index: Towards verification-free query processing on graph databases. SIGMOD, 2007.
3. S. Zhang, M. Hu, and J. Yang. Treepi: A novel graph indexing method. ICDE, pp.966–975,2007.
4. P. Zhao, J. X. Yu, and P. S. Yu. Graph indexing: tree + delta <= graph. VLDB, 2007, pp. 938– 949.
5. X. Yan, P. S. Yu, and J. Han. Graph indexing: a frequent structure-based approach. SIGMOD, 2004.
6. K. Vivekanandan, A. Pankaj Moses Monickaraj, D. Ramya Chithra. Graph Mining Sub Domains and a Framework for Indexing – A Graphical Approach. International Journal of Advanced Computer Science and Applications, Vol. 4, No.4, 2013.
7. Dayu Yuan, Prasenjit Mitra , Huiwen Yu, C. Lee Giles. Iterative Graph Feature Mining for Graph Indexing. IEEE 28th International Conference on Data Engineering, 2012.
8. Xifeng Yan Jiawei Han, gSpan. Graph-Based Substructure Pattern Mining, ICDM 2003. Proceedings, 2002.
9. X. Van, P. S. Yu, and J. Han. Graph indexing: a frequent structure-based approach. Proc. of the ACM SIGMOD international conference on Management of data, pages 335-346, 2004.
10. S. Nijssen and J. N. Kok. The gaston tool for frequent subgraph mining. Electronic Notes in Theoretical Computer Science, Vol. 127, No. 1, pp. 77 – 87, 2005.
11. S. Ranu and A. K. Singh. Graphsig: A scalable approach to mining significant subgraphs in large graph databases. ICDE ’09, 2009, pp. 844–855.