International Journal of Application or Innovation in Engineering & Management (IJAIEM)
Web Site: www.ijaiem.org Email: editor@ijaiem.org
Volume 3, Issue 10, October 2014 **ISSN 2319 - 4847**

A Study to Handle Dynamic Graph Partitioning

Miss. Rupali Jagdale[1], Prof. S. M. Kamalapur[2]

[1] K. K. Wagh Institute of Engineering Education & Research, Department of Computer Engineering,
Savitribai Phule Pune University
Nashik,India
[2] K. K. Wagh Institute of Engineering Education & Research, Department of Computer Engineering,
Savitribai Phule Pune University
Nashik,India

ABSTRACT

Large dynamic graphs are popular and are used in areas like social sites, bio-informatics etc. Huge memory is required to load large graph. Therefore partitioning of large graphs is important in graph analysis. Dynamic updating of the graph can be done with different sub graphs of a large dynamic graph. Different methods of partitioning worked on static graphs. New methods of partitioning can be achieved through combinations of available methods. The proposed work will analyze the large dynamic graph by partitioning it. Also the system will update graph when multiple nodes and edges will be inserted or deleted at runtime. Certain works also focus on generating overlapped nodes in sub graphs. A method to represent large graph in abstract view is also be proposed.

Keywords:- edge insertion, graph partitions, large dynamic graphs, sub-graph

1.INTRODUCTION

Large graphs contain thousands of nodes and edges. They are much complicated as nodes and edges are connected in complex manner. Dynamic graphs are changing with time as different nodes and edges go on adding in graphs. As large dynamic graphs are of compound data structure, such graphs require too much processing of networks and knowledge of a structure of the graph. Loading these graphs with common hardware is quite difficult. Memory usage is more to load such graphs. It is very complicated to comment on accurate size and structure of a large graph as it changes periodically. Large graph analysis starts with separation of the corresponding graph into many small parts called as sub graph. As whole graph cannot be loaded into memory for processing at particular time one must make partitions of Large Graphs. Once graphs are partitioned to analyze, certain operations like updating graphs after node/edge deletion/insertion can be done. These methods of updating large dynamic graphs also focus on sub graphs, that any node/edge insertion or deletion in sub graph should also be reflected in main large dynamic graph. Graph partitioning problems are of NP-hard problem type. Heuristics and approximation algorithms can solve these problems related with partitioning. For tree and grids there are no approximation algorithms that they can solve partitioning problem. Important applications of graph partitioning are scientific computing and task scheduling in multi-processor systems. Recently, the graph partitioning is becoming important due to its applications for clustering and finding the cliques in pathological, social and biological networks. Sparse Linear Systems and Partial differentiation equations can be solved using graph partitioning methods. The paper is organized as follows: Section 1 gives Introduction about Dynamic Graph partitioning. Section 2 introduces related works. Section 3 explains proposed system working. Section 4 Datasets used for graph partitioning. Section 5 concludes the paper.

2.RELATED WORK

Different works focused on the graph partitioning. Static graph partitioning problem is solved by using different algorithms of simple graph partitioning. A try of decreasing memory usage is achieved with some methods [7]. Yet the large dynamic graph partitioning i.e. finding different sub graphs of large graphs is under development [19]. Updating large dynamic graphs as their nodes or edges appear or disappear in network with time is one of the proposed working to be completed [18].

2.1Graph Partitioning Methods

Graph partitioning is important to analyze different large dynamic graphs into their smaller forms. Good graph partitioning is one in which smaller number of edges are there in sub graphs i.e. sub graphs with maximum nodes and less edges. Uniform graph partitioning is a graph partitioning that consists of dividing a graph into sub graphs, where the sub graphs are of nearly the same size and there are fewer connections between edges. There are different methods which can help in partitioning of graphs Local and Global partitioning. The Kernighan–Lin algorithm[14] and Fiduccia-Mattheyses[1] algorithm are local methods. Whereas the global methods rely on properties of the entire graph not on the initial partition of graphs. Spectral partitioning is method of Global partitioning.

2.1.1The Kernighan–Lin algorithm

The Kernighan-Lin Algorithm is older one but it is frequently found in use combined with other methods. This method is an optimization of a benefit function Q, which predicts the difference between the number of edges in the modules and the number of edges lying in between them. The starting point of graph is the starting partition of the graph in two sub graphs of the predefined size: Initial partition can be a random partition of the graph structure. Subsets having same number of nodes are interchanged any sub set may have only one vertex. Some swapping are also arranged to decrease the Q function. After swapping largest Q value is selected and used as new starting point for new iteration. The Kernighan-Lin algorithm is fast, complexity is $O(n^2 \log n)$ (n = number of vertices), Constant number of iterations should occur there. In sparse graphs, quite different heuristic permits for lower complexity as $O(n^2)$. Initial configuration is important for this kind of partitioning. This algorithm has been extended [2] to extract partitions in any number of sub graphs; however the run-time and storage costs increase rapidly with the number of clusters. This is not for community detection as it does not provide input in the form of sub graphs. Also unknown sizes of sub graphs can create problem.

2.1.2Spectral Bisection Method

Spectral Bisection is used when entire graph partitioning is to be performed. This method uses Adjacency matrix and diagonal Matrix. Repeated bisection method is used to get the graph partitioning yet satisfactory results are difficult to achieve. Drawback of this method is that minimum graph sizes or cut points are unknown [4]. Minimum cut partitioning fails if the number of communities gets partitioned, when partition sizes are unknown. Optimizing cut size of group sizes puts all vertices in the same group. Cut size may be the wrong thing to minimize when good divisions are not with small number of edges between communities.

2.1.3Multilevel Partitioning

Multilevel partitioning is frequently solved with repetitive bisection method. Graph is partitioned into two parts consisting same number of nodes. Again these two sub graphs are partitioned into their sub graphs by dividing nodes into same two parts. The graph *G* is first grained to a few hundred vertices then a bisection of this very smaller graph is computed, and then this partitioning is projected back to the original graph by periodically cleansing the partitioning [6].

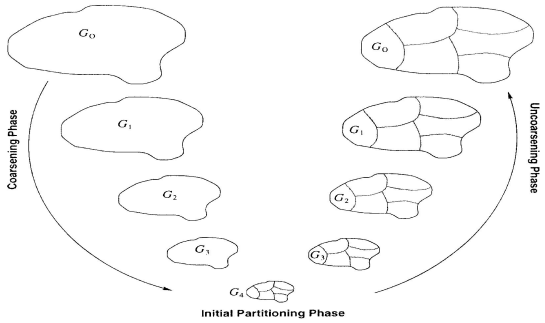

Fig. 1 Multilevel Partitioning [3]

2.1.4Hierarchical Partitioning

Hierarchical Partitioning provides the abstractions of the graphs when we have to see full graph. Hierarchical presentation with tree structure gives the abstraction of parts of large graphs [20]. Loading graph in tree structure and then visualizing its different components are shown here. Algorithms to load nodes, designing basic tree structures and some partitioning methods are used. Many of the methods do not integrate the relation between nodes after hierarchy generation. They do not use proper data structure hence they are limited to main memory only. Issues of managing graph hierarchies efficiently are not solved.

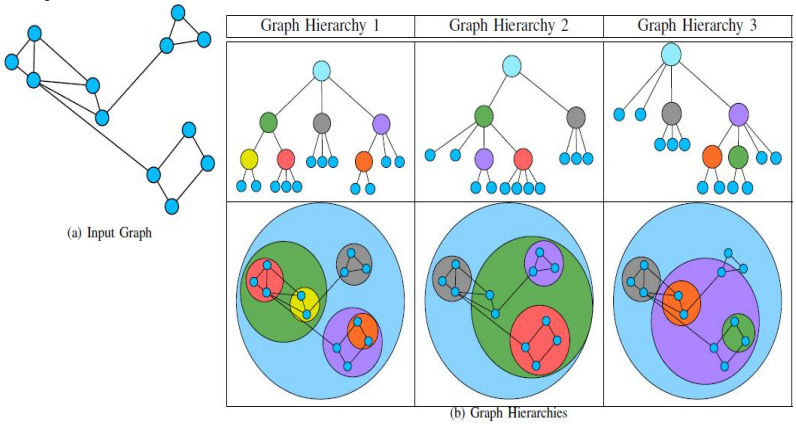

Fig.2 Graph Hierarchy and tree structure [20]

Some graph partitioning methods express multi-criteria optimization problem that can be solved by above mentioned local methods. Game theoretic framework is the best example where each node makes a decision on the partition it selects

2.2Graph Partitioning in GMine System

Jose F. Rodrigues et. al. [18] has worked on analyzing graphs. Graph analyzing includes methods as Graph Partitioning, Graph Summarization.

i.Graph Partitioning:

The Graph Partitioning problem for graph G, having E edges and V vertices. It is possible to decompose G into smaller components with specific properties. It is also called as K-way partitioning which divides the vertex set into k-sub graphs. Graph partitioning is the technique that divides graph into uniform sub graphs.

ii.Graph Summarization

Once the large graph is partitioned into small sub graphs some technique is needed to analyze these sub graphs. center piece sub graph is graph summarization method which locally inspect the sub graph. A center piece sub graph (CEPS) contains the collection of paths connecting a subset of nodes of interest. It can discover a collection of paths rather than a single path. CEPS [21] method uses Random Walk with Restart (RWR) to calculate an importance score between graph nodes. Random Walk refers to stochastic process where the position of entity in a given time depends on its position at some previous time. Another approach MING Approach[22] extends center piece subgraph idea to disk resident graphs and to the entity relationship database context providing the I- Rank measure to capture the informativeness of related nodes.

iii. Graph Visualization

Graph visualization technique provides a visual environment or tool to visualize large graphs using hierarchical representation, clustering or animation

2.3 Need of the proposed system

Dynamic Graph partitioning is an efficient way to analyze the large graphs. This reduces the overall memory usage to load large graphs. Overlapping of sub graph nodes is recognized, to notice repetition of nodes in sub graphs. This can help to get different nodes and their relations with other nodes in different sub graphs. Deletion and insertion of nodes or edges is also analyzed to update the graph dynamically.

3 PROPOSED SYSTEM

3.1 Graph Partitioning

Large Dynamic Graph Partitioning is used to partition graph into sub graphs. In real systems each node after partition belongs to a sub graph. Loading large graph in memory becomes difficult. So we have to divide graph into partitions i.e. sub graphs.

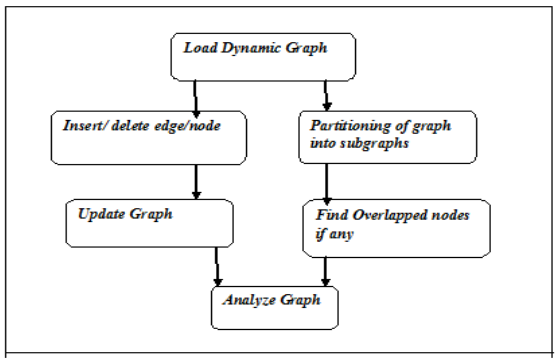

Fig. 3 Working of the proposed system

Steps for partitioning the graph:[19]

1. Read the dataset D
2. Calculate set of adjacent vertices of each vertex available in dataset (Dependent Sets)
3. Calculate size of dependent sets and find limits of dependent sets
4. Calculate partitions by considering largest set first till all vertices are covered
5. Store partitions and dependent sets

3.2 Updating Large Dynamic Graphs

A simple implementation is to recompute the number of nodes/edges after each insertion/deletion methods. Such simple implementation is costly when the graph is very large. A new efficient algorithm can help to update a dynamic graph with new node/edges or removed nodes/edges. Only particular nodes will need to update their degrees when the graph is changed by inserting/deleting an edge. The complexity of algorithm is independent of the graph size [18]. Finally,

wide experiments can be done over both real-world large graphs and different datasets, and the results will express the achievements of graph updations.

4. DATASETS

Some real world datasets are as follows, Wikipedia is a massive static social network dataset which is very large. Such database cannot be completely accessed in internal memory. Hence different nodes of these large graphs can be loaded in memory. The datasets like Flickr and Youtube are real world dynamic social network datasets where the temporal information of each edge is available.

5. CONCLUSION

The existing graph partitioning methods requires excessive processing. Uniform Graph partitioning method is not flexible for large graph. Multilevel Graph partitioning cannot be used for Dynamic Graph Partitioning. GMine System and analysis of large dynamic graph has explained tree structures formations to partition graph, but tree structures don't have specific data structure. Overlapped partitions of static graphs are analyzed in some methods. The proposed method addresses the issue of incomplete main memory by partitioning the large graph and storing the partitions on the disk.

REFERENCES

[1] B.W. Kernighan, S. Lin, An efficient heuristic procedure for partitioning graphs, Bell Syst. Tech. J. 49 (1970) 291_307.
[2] P.R. Suaris, G. Kedem, An algorithm for quadric-section and its application to standard cell placement, IEEE Trans. Circuits Syst. 35 (1988) 294_303.
[3] G. Karypis and V. Kumar, "Multilevel Graph Partitioning Schemes," Proc. IEEE/ACM Conf. Parallel Processing, pp. 113-122, 1995.
[4] A. Pothen, Graph partitioning algorithms with applications to scientific computing, Tech. rep., Norfolk, VA, USA, 1997.
[5] P. Eades and M.L. Huang, "Navigating Clustered Graphs Using Force-Directed Methods," Graph Algorithms and Applications, vol. 4, pp. 157-181, 2000.
[6] E. Dahlhaus, J. Gustedt, and R.M. McConnell, "Efficient and Practical Algorithms for Sequential Modular Decomposition," J. Algorithms, vol. 41, pp. 360-387, 2001
[7] J.S. Vitter, "External Memory Algorithms and Data Structures: Dealing with Massive Data," ACM Computing Survey, vol. 33, no. 2, pp. 209-271, 2001.
[8] I. Finocchi, "Hierarchical Decompositions for Visualizing Large Graphs," PhD thesis, Univ. of Rome, 2002.
[9] E.R. Gansner, Y. Koren, and S.C. North, "Topological Fisheye Views for Visualizing Large Graphs," IEEE Trans. Visualization and Computer Graphics, vol. 11, no. 4, pp. 457-468, July/Aug. 2005.
[10] C. Papadopoulos and C. Voglis, "Drawing Graphs Using Modular Decomposition," Proc. 13th Int'l Conf. Graph Drawing, pp. 343-354, 2005
[11] J.F. Rodrigues Jr., A.J.M. Traina, C. Faloutsos, and C. Traina Jr., "Supergraph Visualization," Proc. IEEE Eighth Int'l Symp. Multimedia (ISM), pp. 227-234, 2006
[12] J. Abello, F. van Ham, and N. Krishnan, "Ask-Graphview: A Large Scale Graph Visualization System," IEEE Trans. Visualization and Computer Graphics, vol. 12, no. 5, pp. 669-676, Sept./Oct. 2006.
[13] M. Raitner, Book Efficient Visual Navigation - A Study by the Example of Hierarchically Structured Graphs. VDM Verlag, 2007.
[14] S. Fortunato, "Community Detection in Graphs," Physics Reports, vol. 486, pp. 75-174, 2010.
[15] N. Zhang, Y. Tian, and J.M. Patel, "Discovery-Driven Graph Summarization," Proc. IEEE 26th Int'l Conf. Data Eng. (ICDE), pp. 880-891, 2010
[16] D. Archambault, T. Munzner, and D. Auber, "Tugging Graphs Faster: Efficiently Modifying Path-Preserving Hierarchies for Browsing Paths," IEEE Trans. Visualization and Computer Graphics, vol. 17, no. 3, pp. 276-289, Mar. 2011
[17] Efficient Core Maintenance in Large Dynamic Graphs Rong-Hua Li, Jeffrey Xu Yu and Rui Mao Sept. 2013
[18] "Large Graph Analysis in the GMine System" Jose F. Rodrigues Jr., Member, IEEE, Hanghang Tong, Member, IEEE, Jia-Yu Pan, Agma J.M. Traina, Senior Member, IEEE, Caetano Traina Jr., Senior Member, IEEE, and Christos Faloutsos, Member, IEEE VOL. 25
[19] "A Partitioning Method for Large Graph Analysis", Miss. Shital Deshmukh, Prof. S. M. Kamalapur, University Of Pune, ISSN-2319-4847
[20] D. Archambault, T. Munzner, and D. Auber, "Grouseflocks: Steerable Exploration of Graph Hierarchy Space," IEEE Trans. Visualization and Computer Graphics, vol. 14, no. 4, pp. 900-913,
[21] H. Tong and C. Faloutsos, "Center-Piece Subgraphs: Problem Definition and Fast Solutions," Proc. 12th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD), pp. 404-413, 2006.
[22] G. Kasneci, S. Elbassuoni, and G. Weikum, "Ming: Mining Informative Entity Relationship Subgraphs," Proc. 18th ACM Conf. Information and Knowledge Management (IKM), pp. 1653- 1656, 2009.

AUTHOR

Miss. Rupali J. Jagdale pursued bachelor of Engineering Degree in Information Technology From KB Thakare College Of Engineering, Nashik, Maharashtra, India (Pune University), now pursuing Post Graduation degree in Computer Engineering from K. K. Wagh Institute of Engineering Education & Reasearch ,Nashik, Savitribai Phule Pune University, Maharashtra, India.

Prof. S. M. Kamalpur working as Associate Professor in Department of Computer Engineering, K. K. Wagh Institute of Engineering Education and Research, Nashik ,Savitribai Phule Pune University, Maharashtra, India