ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Monthly Peer Reviewed Open Access International e-Journal

A graph analysis method with a plenary hierarchical depiction of graphs designed for comprehensive functionalities and interactive visualization

Hareesh Dharmapuri
M.Tech Student,
Department of CSE,
Adam's Engineering College.

B.Simmi
Assoc.Prof & Head of The Dept.
Department of CSE,
Adam's Engineering College.

Abstract:

In mathematics, and more specifically in graph theory, a graph is a representation of a set of objects where some pairs of objects are connected by links. The interconnected objects are represented by mathematical abstractions called vertices, and the links that connect some pairs of vertices are called edges. Typically, a graph is depicted in diagrammatic form as a set of dots for the vertices, joined by lines or curves for the edges. Graphs are one of the objects of study in discrete mathematics. Graph mining is a special case of structured data mining. The growth of the use of semi-structured data has created new opportunities for data mining, which has traditionally been concerned with tabular data sets, reflecting the strong association between data mining and relational databases.

Much of the world's interesting and mineable data does not easily fold into relational databases, though a generation of software engineers have been trained to believe this was the only way to handle data, and data mining algorithms have generally been developed only to cope with tabular data. Due to this graph mining has become a vital content of research. The increasing size of computer graphs is a challenge to mining algorithms. This mining may contain various operations performed on graphs. Graph visualization and summarization are two important operations among them. This paper concentrates on issues of organizing, visualizing and summarizing large graphs.

Keywords:

Graphs, Graph Mining, Graph Visualization, Graph Summarization, hypergraph, Data, Data Anylysis.

Introduction:

In the most common sense of the term, a graph is an ordered pair G = (V, E) comprising a set V of vertices or nodes together with a set E of edges or links, which are 2-element subsets of V (i.e., an edge is related with two vertices, and the relation is represented as an unordered pair of the vertices with respect to the particular edge). To avoid ambiguity, this type of graph may be described precisely as undirected and simple. The vertices belonging to an edge are called the ends, endpoints, or end vertices of the edge.

A vertex may exist in a graph and not belong to an edge. V and E are usually taken to be finite, and many of the well-known results are not true (or are rather different) for infinite graphs because many of the arguments fail in the infinite case. The order of a graph is (the number of vertices). A graph's size is , the number of edges. The degree of a vertex is the number of edges that connect to it, where an edge that connects to the vertex at both ends (a loop) is counted twice. For an edge {u, v}, graph theorists usually use the somewhat shorter notation uv.

XML, being the most frequent way of representing semi-structured data, is able to represent both tabular data and arbitrary trees. Any particular representation of data to be exchanged between two applications in XML is normally described by a Schema often written in XSD. Practical examples of such Schemata, <!- (plural of schema) -> for instance NewsML, are normally very sophisticated, containing multiple optional subtrees, used for representing special case data. Frequently around 90% of a Schema is concerned with the definition of these optional data items and sub-trees.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Monthly Peer Reviewed Open Access International e-Journal

Messages and data, therefore, that are transmitted or encoded using XML and that conform to the same Schema are liable to contain very different data depending on what is being transmitted. Such data presents large problems for conventional data mining. Two messages that conform to the same Schema may have little data in common. Building a training set from such data means that if one were to try to format it as tabular data for conventional data mining, large sections of the tables would or could be empty.

There is a tacit assumption made in the design of most data mining algorithms that the data presented will be complete. The other necessity is that the actual mining algorithms employed, whether supervised or unsupervised, must be able to handle sparse data. Namely, machine learning algorithms perform badly with incomplete data sets where only part of the information is supplied. For instance methods based on neural networks.[citation needed] or Ross Quinlan's ID3 algorithm.[citation needed] are highly accurate with good and representative samples of the problem, but perform badly with biased data. Most of times better model presentation with more careful and unbiased representation of input and output is enough. A particularly relevant area where finding the appropriate structure and model is the key issue is text mining.

XPath is the standard mechanism used to refer to nodes and data items within XML. It has similarities to standard techniques for navigating directory hierarchies used in operating systems user interfaces. To data and structure mine XML data of any form, at least two extensions are required to conventional data mining. These are the ability to associate an XPath statement with any data pattern and sub statements with each data node in the data pattern, and the ability to mine the presence and count of any node or set of nodes within the document.

As an example, if one were to represent a family tree in XML, using these extensions one could create a data set containing all the individuals in the tree, data items such as name and age at death, and counts of related nodes, such as number of children. More sophisticated searches could extract data such as grandparents' lifespans etc. The addition of these data types related to the structure of a document or message facilitates structure mining.

Literature Survey:

In the literature survey we are going to discuss Large Graph Analysis in the GMine System: Below in literature we are discussing some of them.J. Abello, F. van Ham, & N. Krishnan, in this paper describe ASK-GraphView, a node-link-based graph visualization system that allows clustering and interactive navigation of large graphs, ranging in size up to 16 million edges. The system uses a scalable architecture and a series of increasingly sophisticated clustering algorithms to construct a hierarchy on an arbitrary, weighted undirected input graph

D. Archambault, T. Munzner, & D. Auber in this paper several previous systems allow users to interactively explore a large input graph through cuts of a superimposed hierarchy. This hierarchy is often created using clustering algorithms or topological features present in the graph. By allowing users to see several different possible hierarchies on the same graph, it allows users to investigate hierarchy space instead of a single, fixed hierarchy.

D. Archambault, T. Munzner, & D. Auber many graph visualization systems use graph hierarchies to organize a large input graph into logical components. These approaches detect features globally in the data and place these features inside levels of a hierarchy. However, this feature detection is a global process and does not consider nodes of the graph near a feature of interest.

V. Batagelj, W. Didimo, G. Liotta, P. Palladino, & M. Patrignani, many different approaches have been proposed for the challenging problem of visually analyzing large networks. In this paper, we propose a new clustering way whose goal is that of producing both intracluster graphs and intercluster graph with desired topological properties. We formalize this concept in the (X,Y) - clustering framework, where Y is the class that assign the desired topological properties of intracluster graphs and X is the class that defines the desired topological properties of the intercluster graph.

Graph Hierarchical Presentation: Although many works implicitly define the hierarchical clustering of graphs—as in the work of Eades and Feng , Most of them do not touch the issue of how such arrangements deal with scalability and processing by means of a well-defined data structure. Batagelj et al.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Monthly Peer Reviewed Open Access International e-Journal

Proposed System:

The research done on large graph analysis has separately applied graph partitioning and graph summarization methods. The need is to combine these two approaches in one to reduce the computational cost of graph analysis. Graph Partitioning using Dependency Sets The proposed graph partitioning method consists of following steps:

1. Read & parse the input data set D
2. Calculate the sets of adjacent vertices for every vertex from input data set. These sets are called as dependent sets
3. Calculate the size of each dependent set , process and analyze the sets to calculate threshold value of number partitions
4. Calculate the partitions for sets by considering largest set first till all the vertices of data set does not get covered in any of the partition.
5. Store these partitions and dependent sets on the disk

The proposed method in this paper uses the same concept for tree structure formation but with different construction approach which consist of following steps:

1. Find out one vertex of each partition having maximum outgoing degree which is called as Leaf Super node i.e one Leaf Super node represents one partition.
2.Create required number of Super Nodes and Open Nodes which will be used as internal nodes
3.Create a root node first which is called as Super Graph then construct a tree by connecting Leaf Super Nodes directly to Super Graph if there are only two partitions or to internal Super Nodes to balance the tree.
4. Add Open Nodes and external edges to handle edge cuts due to partitioning in tree construction. Once the Graph – Tree is constructed only this hierarchical structure is kept in memory whereas the corresponding partitions are stored on the disk. When user wants to process any partition it will be bring into memory and after processing it will be store on disk. This approach solves the problem of limited main memory.

Finally the system represents graph tree which is an abstract representation of large graph. Once the tree is constructed then user can update any partition whenever required.

Conclusion:

The main issue in large graph analysis is to decompose it into sub graph. The existing graph portioning methods requires excessive processing and some are not scalable for large graph. The proposed method addresses the issue of limited main memory by partitioning the large graph and storing the partitions on the disk.

The graph mining method is based on the clustering, decision tree approaches, classifications, which are fundamentals of data mining. The visualization of graphs is depends on efficient method graph mining only, therefore in paper we have discussed more on graph mining methods and work on large graph representation using the efficient method and frameworks.

References:

[1] Abello, F. van Ham, & 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.

[2] A.L. Buchsbaum and J.R. Westbrook, "Maintaining Hierarchical Graph Views," Proc. ACM-SIAM Symp. Discrete Algorithms, pp. 566-575, 2000

[3] Batagelj, W. Didimo, G. Liotta, P. Palladino, & M. Patrignani, "Visual Analysis of Large Graphs Using (x,y)-Clustering and Hybrid Visualizations," Proc. IEEE Pacific Visualization Symp. (PacificVis), pp. 209-216, 2010.

[4] B.B. Dalvi, M. Kshirsagar,& S. Sudarshan, "Keyword Search on External Memory Data Graphs," Proc. VLDB Endowment, vol. 1, pp. 1189-1204, 2008.

[5] Batagelj, W. Didimo, G. Liotta, P. Palladino,& M. Patrignani, "Visual Analysis of Large Graphs Using (x,y)-Clustering and Hybrid Visualizations," Proc. IEEE Pacific Visualization Symp. (PacificVis), pp. 209-216, 2010.

[6] C. Faloutsos, K.S. McCurley, and A. Tomkins, "Fast Discovery of Connection Subgraphs," Proc. ACM 10th Int'l Conf. Knowledge Discovery and Data Mining (SIGKDD), pp. 118-127, 2004.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Monthly Peer Reviewed Open Access International e-Journal

[7] D Archambault, T. Munzner, & D. Auber, "Grouse flocks: Steerable Exploration of Graph Hierarchy Space," IEEE Trans. Visualization and Computer Graphics, volume 14, no. 4, pp. 900-913, July/Aug. 2008.

[8] D. Archambault, T. Munzner, & D. Auber, "Tugging Graphs Faster: Efficiently Modifying Path-Preserving Hierarchies for Browsing Paths," IEEE Trans. Visualization and Computer Graphics, volume 17, no. 3, pp. 276-289, March. 2011.

[9] D. Harel and Y. Koren, "Graph Drawing by HighDimensional Embedding," Proc. Revised Papers from 10th Int'l Symp. Graph Drawing, pp. 207-219, 2002.

[10]D. Auber, Y. Chiricota, F. Jourdan, & G. Melanc¸on, "Multiscale Visualization of Small World Networks," Proc. IEEE Ninth Conf. Information Visualization (InfoVis), pp. 75-81, 2003.