International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Impact Factor (2012): 3.358

Graph Presentation in GMine System using Efficient Algorithm

Shafali Gupta[1], Ulka Panchal[2]

[1, 2] Pune University, RMD-Sinhgad School of Engineering, Warje, Pune, India

Abstract: ***Thousands of existing applications nodes and edges on the order of hundreds of millions of graphically produced. To take Benefits such as illustrations, patterns, outliers and communities should be able to find these functions are performed in a Interactive environment where human expertise can guide procedure. larger graphs, however, there are some challenges: Highly prohibitive processing requirements, and drawing a hundred-thousand nodes results in cluttered images difficult to Understand these problems, we would like an innovative structure trees suited to any kind of Visual design graph. GMine integrates 1) underlines a representation organized hierarchies Super Graph of Division and conceptual graph-tree; And 2) a graph representation summarization method tracing our CEPS deals with connection problem Sub linear complexity, with a graph to a single node or nodes of a hierarchy, a neighborhood group to allow for an understanding of the aspects A proof of concept with one click., a system in which large graphs can be GMine the Visual environment is instantiated Globally and locally. 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.***

Keywords: Graph analysis system, graph representation, data structures, graph mining, graph visualization.

1. Introduction

Large graphs are common in real-life settings: web graphs, computer communication graphs, recommendation systems, social networks, bipartite graphs of weblogs, to name a few. To find patterns in a large graph, it is desirable to compute, visualize, interact and mine it. However, dealing with graphs on the order of hundreds of thousands of nodes and millions of edges brings some problems: the excessive processing requirements are prohibitive, and drawing hundred-thousand nodes results in cluttered images that are hard to comprehend. The visualization of large graphs is accompanied by effective interaction techniques, in particular, in cases when the whole graph is too complex or large to be visualized in one static view.

The contribution of this work is the integration of methodologies that address the problems discussed above. We introduce a novel representation for graph hierarchies that extend those of previous works, leading to a model more suitable for presentation and computation.

Representation and processing: What are the adjacencies of a given graph node considering the entire graph, and not only its particular partition? Mining: Given a subset of nodes in the graph, what is the induced sub graph that best summarizes the relationships of this subset? . Visualization: How do we see through the levels of the graph hierarchy? . Interaction: How do we perform all these tasks efficiently and intuitively? It is our contention that a system that presents the original graph concomitant to its hierarchical version must meet all these requirements. Therefore, we seek for a new representation for graph hierarchies, different from previous works in which the graph hierarchy is "stagnant" and cannot answer questions about the relationships between nodes at different groups, and neither between groups at different partitions of the hierarchy. Besides the capability of globally analyzing large graphs, our system is complemented with the possibility of locally analyzing a sub graph that is part of a larger graph hierarchy.

Thus in this paper we introduce a Visual design graph, organized hierarchies Super Graph of Division and conceptual graph-tree representation by GMine integrates. Super Graphs are formalizes the essentials of the Graph-Tree and the Graph-Tree incorporates the Super -Graph abstraction.

In next section II we are presenting the literature survey over the various methods security at data sharing systems. In section III, the proposed approach and its system block diagram is depicted. In section IV we are presenting the current state of implementation and results achieved. Finally conclusion and future work is predicted in section V.

2. 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, [1] 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 [2] 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 [3] 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, [5] 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 [12], 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. [7],

3. Proposed Approach Framework and Design

3.1 Problem Definition

The large graph problem has been treated through graph hierarchies, according to which a graph is recursively broken to define a tree of sets of partitions. However, previous efforts on this matter fail on the task of integrating the information from multiple partitions, disregarding mining techniques to fine inspect each sub graph.

3.2 Proposed Architecture and Design

3.2.1 Super graphs and the graph tree

Our first contribution is an original formalization of graph hierarchies engineered to support processing and presentation. We define the Super Graphs concept, an abstraction that converges to an implementation model we have named Graph-Tree. While Super Graphs formalize the essentials of the Graph-Tree, the Graph-Tree incorporates the Super- Graph abstraction. The closest work to the ideas of Super Graph and Graph Tree was proposed by Abello et al. [1].

Their work formalizes a hierarchy tree, whose data structure is based on what they name ant chains sets of nodes such that no two nodes are ancestors of one another. Their formalization parallels with ours by the concept of macro similar to the terminology super, used along this work. The originality of our approach is that the graph hierarchy is not available only for visual interaction; it can be used for processing at any level of the tree just as if the original graph was a thorough plain representation.

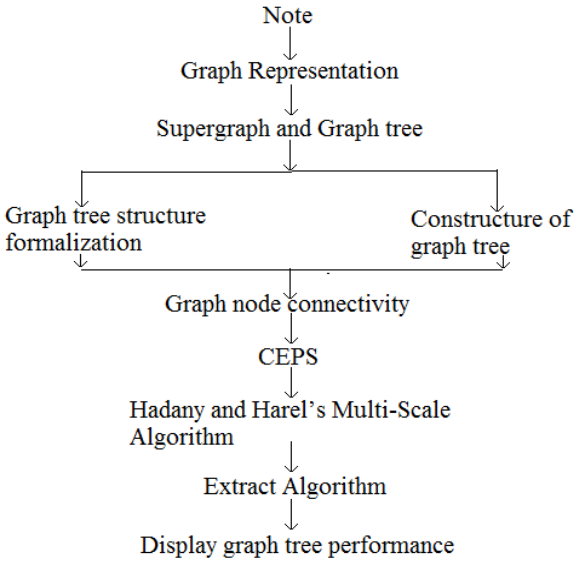

Figure 1: Proposed System

3.2.2 Graph-Tree Structure Formalization

For the purpose of formalizing the Graph-Tree structure,1 following we define a set of abstractions that encompass its engineering, starting by the notion of Super Graph. The underlying data beneath a Super Graph are a graph G = fV ;Eg—with jV j nodes and jEj edges—but a Super Graph presents a different abstract structure. It is based on the observation that the entities in a graph can be grouped according to the relationships that they define. This concept allows us to work with a graph as a set of partitions hierarchically defined. In the following, we define the constituents of a Super Graph,

3.2.3Basic Definitions of the Super Graph:

The choice for a specific graph partitioning is independent of the Graph-Tree methodology. We perform a sequence of recursive partitioning. Each recursion generates k partitions to form the next level of the tree, a process that repeats until we get the desired number of h hierarchy levels. At the end of the process, references to the sub graphs are kept at the leaves. From the storage point of view, the tree structure is kept on main memory, while the sub graphs are kept on disk, being read only when necessary.

a) Filling the Graph-Tree Super Nodes:
After obtaining a hierarchy, it is necessary to fill the Super Nodes of the tree with their Super Edge and open nodes information. In Algorithm 1, the Graph-Tree is recursively traversed bottom-up along its levels.

b) CEPS: Center-Piece Sub graph
Although graph hierarchies can lessen the problem of globally inspecting large graphs, we have found that it is common to reach the bottom of the Graph-Tree and have a sub graph that presents more information than what is desired, in a layout that suffers with node overlapping. In this situation, although the user is able to compute, draw, and interact with the graph nodes of a Leaf Super Node, there might still be too many edges and nodes, preventing examination. This happens naturally, either on large graphs or on moderate to small graphs.To remedy this problem, we benefit from the concept of Center-Piece

Sub graph (CEPS for short) to complement the analytical environment of GMine. A center-piece sub graph contains the collection of paths connecting a subset of graph nodes of interest. Using the CEPS method, a user can specify a set of query graph nodes and GMine will summarize and present their internal relationship through a small (say, with tens of nodes), yet representative connection sub graph.

c) *The "EXTRACT" Algorithm*:
The "EXTRACT" algorithm takes as input the graph G0, the importance/goodness score r(Q; j) on all nodes, and the budget b, and produces as output a small, undirected graph CP. The basic idea is as follows: 1) instead of trying to find an optimal subgraph maximizing g(CP) directly, we decompose it, finding key paths incrementally; 2) by sorting the graph nodes in order, we can quickly find the key paths by dynamic programming in the acyclic graph.

d) *Hadany and Harel's Multi-Scale Algorithm*:
The multi-scale approach proposed by Hadany and Harel [HH99] is an improvement of the force-directed technique, which facilitates the drawing of larger graphs.

3.3 Mathematical Module

We consider graphs $G = (V, E, w)$ where V *and* E denote the set of vertices and edges respectively. w is a non-negative real weight function assigns to each edge in E. $|V|$ and $|E|$ denotes cardinalities of V and E. A subgraph of G is to be induced by a subset U of V, if it consists of the nodes in U and those edges in E that have both endpoints in U. We denote an induced subgraph by G (U). A hierarchy tree TG for graph G is a tree rooted at r whose set of leaves is in one to one correspondence with V. That is, *leaves*(r) $=V$, if we denote a set of descendant leaves of a node u in the tree by *leaves*(u). The set of children of a node u is denoted by *children*(u). Similarly, *parent*(u) is denoted the unique parent of u. A (maximal) antichain A in TG is a (maximal) set of nodes in TG such that no two distinct nodes in A are ancestors of one another.

4. Work Done

In this section we are presenting practical environment.

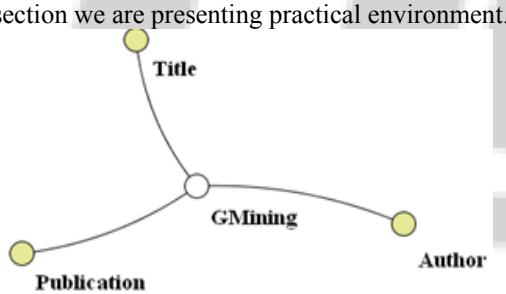

Figure 2: Gmining graph is divided into three parts Title, Publication and Author.

In Figure 1 the input data is represented into the graph and the graph is initially divided into three subgraphs named as Title, Publication and Author.

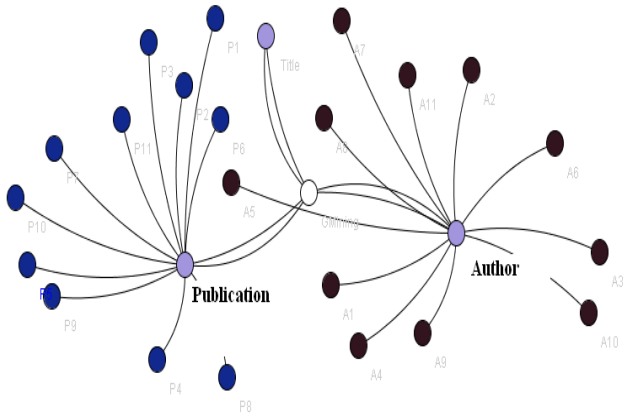

Figure 3: Publication and Author graph is divided into many subgraphs

In figure 2, the publication and Author graph is again divided into many subgraphs which are related to each other.

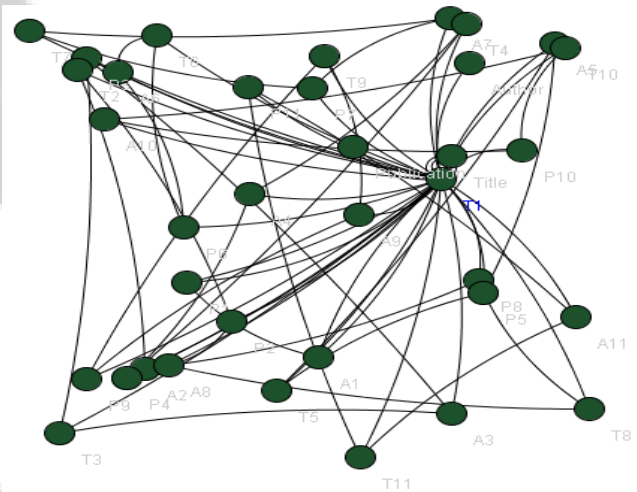

Figure 4: Search result according to Author Name

The figure 4 show the search result for the input data author name, it displays related data to the author name in the graph form. Figure 5 shows the comparison result of the existing system which uses Gmine system without lossless compression technique to represent graph and the proposed system which uses Gmine system with the lossless compression technique to reduce the memory size required to store the graph data.

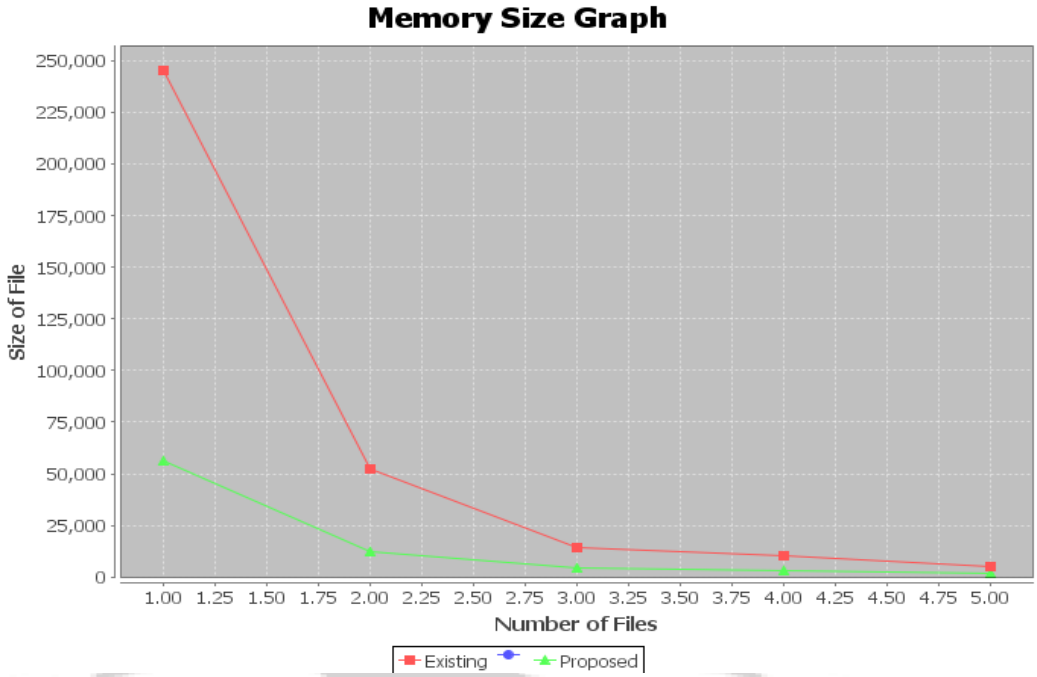

Figure 5: Memory use comparison between existing system and Proposed System

4.1 Hardware and Software Used

a) Hardware Configuration
- Processor - Pentium –IV
- Speed - 1.1 GHz
- RAM - 256 MB (min)
- Hard Disk - 20 GB
- Key Board - Standard Windows Keyboard
- Monitor – SVGA

b) Software Configuration
- Operating System: Windows XP/7/8
- Programming Language: Java
- DATABASE: MY SQL.
- Tools: Net beans.

5. Conclusion and Future Work

GMine presents a system for large view graphs Analysis supports process framework that can GMine Hundreds of thousands of nodes using large graphs with Hierarchical graph partitioning and the interactive summarization, Includes scalability via an innovative contribution Graph formalization processing graph hierarchies for and representation, an innovative connection sub graph Extraction algorithm, and a presentation of the concept of evidence Large graphs. 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.

[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 High-Dimensional 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.

[11] E. Dahlhaus, J. Gustedt, and R.M. McConnell, "Efficient and Practical Algorithms for Sequential Modular Decomposition," J. Algorithms, vol. 41, pp. 360-387, 2001.
[12] E.R. Gansner, Y. Koren, and S.positions for Visualizing Large Graphs," PhD thesis, Univ. of Rome, 2002.
[13] F. van Ham & J.J. van Wijk, "Interactive Visualization of Small World Graphs," Proc. IEEE Symp. Information Visualization (InfoVis), pp. 199-206, 2004.
[14] H. David, "Statecharts: A Visual Formalism for Complex Systems," Science Computer Programming, vol. 8, pp. 231-274, 1987.
[15] M. Gomez-Rodriguez, J. Leskovec, & A. Krause, "Inferring Networks of Diffusion and Influence," Proc. 16th ACM SIGKDD Int'l Conf. Knowledge Discovery & Data Mining, pp. 1019-1028, 2010.
[16] M.L. Huang and Q.V. Nguyen, "A Space Efficient Clustered Visualization of Large Graphs," Proc. Fourth Int'l Conf. Image and Graphics, pp. 920-927, 2007.
[17] P. Vaz de Melo, L. Akoglu, C. Faloutsos, and A. Loureiro, "Surprising Patterns for the Call Duration Distribution of Mobile Phone Users," Proc. European Conf. Machine Learning and Knowledge Discovery in Databases (ECML-PKDD), pp. 354-369, 2010.
[18] P. Eades and Q. Feng, "Multilevel Visualization of Clustered Graphs," Proc. Symp. Graph Drawing, pp. 101-112, 1997.
[19] http://www.research.att.com/groups/infovis/res/legacy_papers/DBLP-journals-jgaa-HarelK02.pdf

Author Profile

Shafali Gupta Asst. Professor in Compute Engineering department RMD-Sinhgad School of Engineering Warje, Pune, Maharashtra, India**.**

Ulka Panchal birth place is Udgir dist. Latur on July 1st 1982, completed engineering in computer engineering branch from SRTMU university Nanded, Maharashtra, India in 2005.