ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 7, July 2013

Analysis of Large Graph Visualization and Summarization

Akshata D. Deore[1], Prof. R. L. Paikrao[2]

ME Computer Engg., AVCOE, Sangamner [1]

Associate Prof. And HOD Computer Engg., AVCOE,Sangamner[2]

Abstract: Graphs are extensively used to represent difficult structures in several scientific and industrial applications. Frequent graphs occurring often in group of graphs are specially used in social networks, computer networks, citation network and many more. 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: Frequent Graphs, Graphs, Graph Mining, Graph Visualization, Graph Summarization

I. INTRODUCTION

A group of nodes connected using links or edges are known as Graph. In general a graph may only be viewed as connected components, but if carefully examined and processed it can be used for various types of applications [1]. Some of the applications are explained below:

- Chemical and Biological Data :

Graphs can be used to represent chemical data efficiently. Graph nodes are used to show atoms and graph edges or links shows bonds between these atoms. Detailed structure of chemicals can be obtained by representing sub-units of the molecules using data and their bonds using links. For example Protein-Protein interaction shows Nodes corresponding to Proteins and edges corresponding to interaction between these proteins. This type of data becomes very important when analyzed to seek out any new drug. Repeating representation of same node allows one to find certain patterns from them. Pattern finding results into subgraphs which in turn leads to give detailed results about specific pattern (chemical or compound).

The same method is used to represent biological data. Nodes as amino acids and edges as a link between them are represented using graphs.

- Computer Network:

To show computer network using a graph is the best way to represent it. Computers are connected to each other in a network to form interconnection. Computers are nodes and their interconnections are edges in this scenario. This type of structure is used to traverse the graph to find out various route, shortest route, etc.

- Web Data:

World Wide Web is a vast collection of web pages connected together. Web data can be exhibited using graphs. Web pages can be shown using graph nodes and edges exhibits links or interconnection between these web pages. In this way, web is generally structured in the form a graph. This graph data is mined to search particular web page, to find patterns in web pages, to personalize the web, etc. Social web site is the best example of web data represented as a graph.

- XML Data:

XML data is naturally represented in a structured format. Queries are nested into one another; therefore this structure can be viewed as a labeled graph. This structured data becomes more complicated when attributes are added to the nodes. Since mining these queries to get required data has become essential, graph mining is widely used in this field.

When such data is represented using graphs, it requires various operations to be performed for gaining interesting results. Due to ability of modelling complex structure, graphs are gaining wide popularity [2]. With this increased demand of analysis of large graph data; graph processing has become operative and valuable area of data mining. The

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 7, July 2013

main aim of graph processing is to explore graphs and to get more information from this structured data [3]. When these graphs are small in nature it becomes easy to work with them. Issues occur when this data becomes bigger and bigger, it increases complexity level of processing and displaying them [4].

There are three main issues for processing these large graphs are given below:

1. A large graph may contain hundreds of nodes and thousands of edges, which are not fitted on a single display screen when we try to visualize it. It results in cluttered structure and becomes difficult to understand [5].
2. Extracting only useful or intended information from it becomes another concern to worry about complex structures [6].
3. When any new information is added to the graph database, the system may process complete database which becomes very time consuming.

To solve this problem the system called Analysis of Large Graph Visualization and Summarization is developed. This system mainly concentrates on the efficient visualization of huge graph database and search only needed data by summarizing.

This paper is organized into sections. Section II explains working methodologies of this system. Results are shown in Section III and this paper is concluded in Section IV.

II. ANALYSIS OF LARGE GRAPH VISUALIZATION AND SUMMARIZATION SYSTEM

This system is divided into three main parts. First part describes the graph visualization system. Graph summarization is explained in second part and lastly dynamic analysis is done in the last part.

A. *Graph Visualization:*

For efficient visualization of large and complicated graph structures SuperGraph [7] concept is used. This concept is developed by Jose F. Rodrigues Jr. et al. This SuperGraph plays an important in construction of the Graph-Tree as named by the authors [7].The concept of SuperGraph believes in collecting the entities having some relationship. Some definitions [7] are given below to make a clear understanding of SuperGraphs and Graph-Tree; these definitions are developed by Jose F. Rodrigues Jr. et al. for "Large Graph Analysis in the GMine System."

Definition 1: [SuperGraph] *Given a finite undirected graph* $G = \{V, E\}$*, with no loops nor parallel edges, a SuperGraph is defined as* $\bar{G} = \{\bar{V}, \bar{V}_l, \bar{E}\}$ *where,* $\bar{V}$ *is a set of SuperNodes* $\bar{v}$*,* $\bar{V}_l$ *is a set of LeafSuperNodes* $\bar{v}_l$ *and* $\bar{E}$ *is a set of SuperEdges* $\bar{e}$*.*

Definition 2: [LeafSuperNode] *Given a subset of graph nodes* $V' \subset V$*, a LeafSuperNode* $\bar{v}_l$ *is defined as the subgraph* $G' = \{V', E'\}$*, where* $E' = \{ (u,v) \mid (u,v) \in E$ *and* $u,v \in V'\}$.

Definition 3: [SuperNode] *A SuperNode* $\bar{v}$ *is recursively defined as a set of* $\bar{V}'$ *of SuperNodes, or LeafSuperNodes,* $\bar{v}_i$ *plus a set of* $\bar{E}$ *of SuperEdges* $\overline{e_{ij}}$*. As follows:*

$$\bar{v} = \{\bar{V}' = \{\overline{v_0}, \overline{v_1}, \dots, \overline{v_{(|\bar{v}'|-1)}}\}\}$$
$$\bar{E}' = \{\overline{e_{ij}} = \{\bar{v}_i, \bar{v}_j \mid \bar{v}_i, \bar{v}_j \subset \bar{V}'\}\}$$

Where, $\bar{v}_i$ *can be either a SuperNode or a LeafSuperNode.*

Definition 4: [SuperEdges] *A SuperEdge represents all edges* $(u, v) \in E$ *that connect graph nodes from a SuperNode* $\bar{v}_i$ *to graph nodes from SuperNode* $\bar{v}_j$ *. A SuperEdge* e_{kk} *for a LeafSuperNode* $\overline{V_{lk}} = \{V_k', E_k'\}$ *holds all the edges that interconnect graph nodes in the LeafSuperNode* v_{lk} *that is, all the edges in* E_k'*. Formally, the SuperEdge between SuperNodes* $\bar{v}_i$ *and* $\bar{v}_j$ *is defined as follows:*

$$SuperEdge(\bar{v}_i, \bar{v}_j) = \overline{e_{ij}} = \{e = (u,v) \mid (u,v) \in E.$$
$$u \in Coverage(v_i) and\ v \in Coverage(v_j)\} = eij\}$$

Definition 5: [External edge] *An edge e is called an external edge of* $\bar{v}$ *if source(e)* $\in$ *Coverage(*$\bar{v}$*) and target(e)* $\notin$ *Coverage(*$\bar{v}$*). The external edge e cannot be resolved within the Coverage of* $\bar{v}$*.*

Definition 6: [Open node] *A graph node v* $\in$ *Coverage (*$\bar{v}$*) is called an open node of* $\bar{v}$ *if there exists an external edge e in the set of external edges of (*$\bar{v}$*) where source (e) = v. We denote the set of all the open nodes of a SuperNode* $\bar{v}$ *as OpenNodes(*$\bar{v}$*).*

To perform visualization of large graphs includes following tasks:

a. Hierarchy Formation:

To form the partitions a recursive employment of k-way graph partition method [8] is done. A recursion generates k partitions every time and forms next level of the hierarchy. Each new partition will form a subgraph. Each new subgraph is then included in the Graph-Tree and its references are stored at the leaves when this process ends. In this way large graph is partitioned according to its relations.

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 7, July 2013

b. Putting information into the Hierarchy:

This process is used to fill the relevant information in the Graph-Tree. This recursive method begins at the low level of the Graph-Tree by filling LeafSuperNode information and goes upwards to find open nodes using external edges. In this way by tracking external edges and open nodes the Graph-Tree is filled.

c. Computing Connectivity of SuperGraph:

When large graphs are partitioned and visualized in this way, the concern of keeping the original graph information arises. To deal with it two methods are used.

Sibling SuperNodes are easily connected for maintaining the original relation of graphs. The task is to connect SuperNodes which are not siblings. For this purpose first all possible connecting edges are found by the Cartesian product of open nodes of two SuperNodes. Then common parent's connecting edges are tracked.

To maintain the relationship of nodes belongs to different entities, GraphNode Connectivity is performed. The GraphNode Connectivity for a graph node is achieved by collecting all the edges connecting that node with others that are not related to SuperNodes.

These two things help to find SuperNode connectivity.

B. *Graph Summarization:*

To achieve summarized graph Center-Piece Subgraph (CEPS) [9] algorithm is used. To find nodes of interest goodness score of the graph nodes is calculated. Random walk with restart method [10] is used to find out goodness score of a single node for a single query. It helps to find the closeness of a node with a single query node. The goodness score of a node can also be calculated for a query set. Once the goodness score is achieved the next step is performed to extract the subgraph. The EXTRACT algorithm accepts the subgraph and the goodness score of all nodes as an input; it also accepts a parameter budget b to keep track of the subsets of nodes. It fist selects a destination node and discovers a new path between source and destination node. Like this new paths are added to a set of subset and duplicate path nodes are merged.

C. *Dynamic Analysis:*

Dynamic analysis states that when any new nodes are added to the graph database, these nodes should be added to the Graph-Tree structure so that they can be processed and analyzed. While adding these nodes to the already formed structure instead of processing the complete structure again, only new nodes are added and processed. This saves the computation time of updating the graph data to be processed. To do this the first the nodes are added to the graph database. Then hierarchy construction of only new nodes is performed to include it into the Graph-Tree. When filling the information to this, external edges are propagated this gets SuperEdges and open nodes. The main task in dynamic processing is forming the connectivity of new nodes with an existing Graph-Tree. To do this GraphNode Connectivity is used. This connects the newly connected group of graphs with an existing Graph-Tree. It is then ready to be processed by then system.

III. RESULTS AND DISCOSSIONS

Here the visualization system is introduced. This system uses Graph-Tree and SuperGraph concepts and produces a visualization of graphs. For this system a dataset of author and publication is used. The authors [6] have provided a dataset for developers called as Digital Bibliography and Library Project (DBLP). The DBLP consists of 315,688 author nodes and 1,659,853 co-authoring edges. For simplicity purpose we have used a similar but simplified dataset. This dataset consists of author and co-author relationship. Nodes represent authors and links represent co-authors, from this graph the publication details can be derived.

The visualization system is shown in fig. 3.1. The white node is the root node of the graph and children nodes are displayed using various colours. The layout of the graph can be arranged using different layout options. This system is able to deal with the scalability issue unlike the work done by Eades and Feng [11], they are not able to handle the scalability problem. Also for dynamic modification this system gives an efficient result which is not provided by Archambault et al. [12]. Thus in this way it provides an interactive environment for graph visualization and summarization.

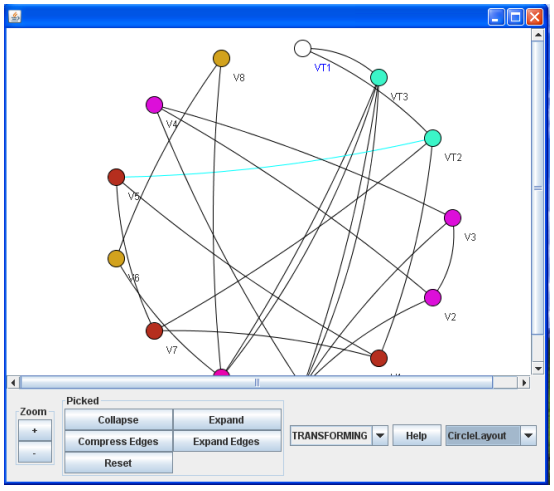

Fig. 3.1 Graph Visualization System

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 7, July 2013

IV. CONCLUSION

The presented system is used for visualizing and summarizing the large graphs. To simplify the large graph structure this system divides the large graphs into different groups to form a hierarchy, fills information to this hierarchy and then connects them to form a complete large graph again. This helps the visualization system to display the large graph in an easy manner. Summarization is done using a CEPS algorithm. This algorithm helps to derive the intended data from a huge and complicated collection of graphs. The system efficiency is improved by providing the dynamic connectivity feature which reduces the time of processing the newly connected nodes and connecting them to the existing graph.

REFERENCES

[1] Christos Faloutsos, *Graph Mining: Laws, Generators and Tools*, CMU
[2] Jiawei Han and MichelineKamber, *Data Mining: Concepts and Techniques*, University of Illinois at Urbana-Champaign
[3] Haixun Wang And Charu C. Aggarwal, *Managing And Mining Graph Data*
[4] F. van Ham and J.J. van Wijk, *Interactive Visualization of Small World Graphs*, Proc. IEEE Symp. Information Visualization (InfoVis), pp. 199-206, 2004.
[5] P. Eades and Q. Feng, *Multilevel Visualization of Clustered Graphs*, Proc. Symp. Graph Drawing, pp. 101-112, 1997.
[6] D. Schaffer, Z. Zuo, S. Greenberg, L. Bartram, J. Dill, S. Dubs, and M. Roseman, *Navigating Hierarchically Clustered Networks through Fisheye and Full-Zoom Methods*, ACM Trans. Computer-Human Interaction, vol. 3, pp. 162-188, 1996.
[7] Jose F. Rodrigues Jr., Hanghang Tong, Jia-Yu Pan, Agma J.M. Traina, Caetano Traina Jr., Christos Faloutsos, *Large Graph Analysis in the GMine System* , IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 25, NO. 1, JANUARY 2013.
[8] G. Karypis and V. Kumar, *Multilevel Graph Partitioning Schemes*, Proc. IEEE/ACM Conf. Parallel Processing, pp. 113-122, 1995.
[9] 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.
[10] Nathan McNew, *Random Walks with Restarts, 3 Examples*, March 3, 2013.
[11] P. Eades and Q. Feng, *Multilevel Visualization of Clustered Graphs*, Proc. Symp. Graph Drawing, pp. 101-112, 1997.
[12] 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.