ModulGraph: Modularity-based Visualization of Massive Graphs

Chenhui Li, George Baciu,* Yunzhe Wang
Department of Computing, The Hong Kong Polytechnic University

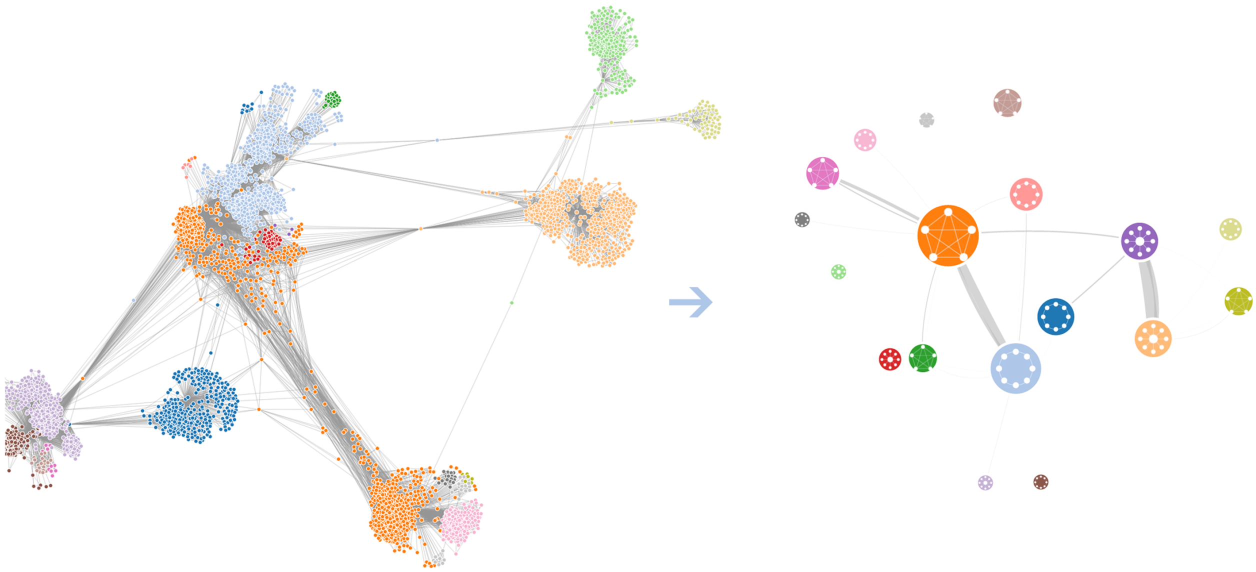


Figure 1: *A comparison between the common graph visualization and ModulGraph-based graph visualization. (left) A simple example of visualizing a graph. (right) The result of ModulGraph.*

Abstract

Large graph visualization has become a dominant problem in multiple big data analytics domains, including media analytics, social network dynamics, resource management in cloud computing environments, air travel, large networks. A practical approach to displaying massive graphs is by partitioning according to well defined domain-dependent attributes. However, graph visualization in the presence of incomplete information is an open challenge in many applications. In order to better visualize and understand patterns in large graphs, local pattern discovery becomes a critical step in deciding the structural components of graph visualization. In this paper, we present a modularity-based graph visualization method, termed as the *ModulGraph*. The ModulGraph is a hierarchical representation that treats a graph as a set of modules. The main objective of this work is to hierarchically detect graph patterns in order to visualize large graph data and adapt the interconnecting structures to potential interactions between local module streams. Our main contribution is a graph visualization method that can flexibly detect the local patterns or substructures, called modules, in large graphs. The second contribution is a hybrid modularity measure. This measures hierarchically the cohesion of the graph at various levels of details. We aggregate clusters of nodes and edges into several modules for the purpose of reducing the overlap on the display. Graph patterns of modules are processed by the ModulGraph system in order to avoid information loss while a sub-graph is represented as a single node. Our experiments show that this method can support large-scale graph visualization for visual media exploration and analysis.

CR Categories: H.5.2 [Information Interfaces and Presentation]: User Interfaces—Evaluation/Methodology, I.6.9 [Visualization]: Information Visualization—Visualization Techniques and Methodologies;

Keywords: Large graph, community detection, modularity, level of detail, information visualization

1 Introduction

Large graph analysis and visualization has gained much interest in the big data analytics community. Graph connectivity patterns are of interest in detecting unexpected features and allow us to discover and isolate points of interest in massive networks of attributed data such as social networks, modern transportation networks, document referencing, academic citations, semantic nets, and many others. However, the effect of graph visualization is often limited to the size of the display. To minimize clutter and noise, it is often desirable to aggregate the components of large graphs into clear modules in order to help users cope with graph size limitations and effectively gain insights of attributed graph data.

In this paper, we present the *ModulGraph*, a flexible graph visualization framework, that can adapt to large-scale graph pattern visualization. The main idea of this paper is to partition the graph

*e-mail:csgeorge@comp.polyu.edy.hk

into several clusters and visualize the relationships between clusters. For example, in social networks, strong clusters represent social communities. By visualizing these clusters, we can often find missing or related information. Hence, we can further design the graph patterns for each cluster and allow users to understand the structure of the graph.

We first address the problem of module-based graph visualization according to *modularity measure*. Second, a graph simplification method is adopted to accelerate module detection. Third, each graph pattern in the module is analyzed to enhance the module visualization. Forth, the visualization of *ModulGraph* is constructed from two parts. The first part is a community graph that indicates the basic structure of the original graph. Another part is graph pattern visualization. We use symbolic signature instead of simple node in the community graph. Our experiments show that *ModulGraph* can effectively transform large graph data into communities .

Our method can also be extended to a dynamic graph visualization that can present the evolution of the graph. In addition, ModulGraph is suitable for aggregating graphs from different time steps into a super-graph in order to explore the whole structure.

This rest paper is organized as follows. Section 2 shows related work of graph visualization. Section 3 presents the community detection method we will select and analyses the graph patterns. The visualization of ModulGraph is presented in this section. Section 4 describes our cases of experiments and discusses the results. Section 5 concludes this work and plan the future work.

2 Related Work

For tracking and understanding large graphs, most of the research topics have been focused on information visualization. Zinsmaier and his colleagues [Zinsmaier et al. 2012] are inspired from the technique of Level-of-Detail in computer graphics discipline and introduced straight-line graph drawing that can be rendered interactively with level of detail in order to visualize large-scale contents. Since the single screen is limited for visualizing the large graph, the solution of multi-screen is adopted by the researcher as shown in [Chae et al. 2012], however the multi-screen approach imposes restrictions on the space and interactions.

Newman [Newman 2006] used modularity to measure the performance of the algorithm on community detection. They converted the problem into an optimization problem. Blondel et al. [Blondel et al. 2008] presented an accelerating method called Louvain for modularity-based community detection. They demonstrated the performance of Louvain on several large graphs.

Some methods of graph simplification are presented such as graph summarization [Tian et al. 2008], HiMap [Shi et al. 2009] and K-Core [Baxter et al. 2012]. Graph summarization method [Tian et al. 2008] allows the user to interactively control the resolution of each aggregation in a large graph. HiMap [Shi et al. 2009] is presented to visualize large-scale social network through hierarchical summarization. Complexity reduction via K-Core [Baxter et al. 2012] is used to remove the nodes with less than k links.

Rosvall [Rosvall and Bergstrom 2008] and Louvain [Blondel et al. 2008] discuss the community aggregation. This aggregates the nodes of a community into a super-node. A machine learning method called *Belief Propagation* [Horng et al. 2011] is adopted to infer the other nodes that may be of interest in order to explore large graphs. Dunne et al. [Dunne and Shneiderman 2013] improve the graph visualization readability through drawing different glyphs. Our work is similar to Dunne et al. [Dunne and Shneiderman 2013], but we focus on modularity-based motif detection.

3 ModulGraph

ModulGraph is a graph that can be viewed as an abstraction of a large graph. We denote the topology of the large graph with $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, where $\mathbf{V} = [\mathbf{v}_0^T, \mathbf{v}_1^T, ..., \mathbf{v}_n^T], (\mathbf{v_i} \in \mathbb{R}^2)$ denotes the vertices and $\mathbf{E}$ denotes the undirected edges. The ModulGraph is defined as $\mathbf{Mg} = (\mathbf{M}, \mathbf{E})$, where $\mathbf{M} = [\mathbf{m}_0^T, \mathbf{m}_1^T, ..., \mathbf{m}_n^T], (\mathbf{m_i} \in \mathbb{R}^2)$ denotes the modules and $\mathbf{E}$ denotes the undirected edges, where each edge has a weight of $w = |E|$. Each $\mathbf{m}$ is summarized as a vector with several features such as the relative position of the module, module type, sub-node count and sub-link count. All of these features can be calculated through the module detection.

3.1 Module Detection

On the whole, each module could be considered as a community in the graph. Hence, the community detection method called *Modularity Classes* as described in [Newman 2006] can be adopted to find the communities from a graph. A further improved method called *Louvain* is presented by Blondel et. al [Blondel et al. 2008], which reduce the calculation of the modularity and make scene of large counts of communities detection. *Modularity Classes* is time-consuming for dealing with large-scale graph and the precision of the *Louvain* is less than previous one. We present a hybrid modularity-based method in order to take the advantages of these two methods. Our method is based on the modularity and is suitable for calculating the modules for a large graph. The process of finding the modules is optimizing the modularity of the linked nodes. The modularity can be defined as [Newman 2006]:

$$\mathrm{Q} = \frac{1}{2m} \sum_{i,j} \left[\mathrm{A}_{mn} - \frac{k_i k_j}{2m} \right] \lambda(s_i, s_j) \tag{1}$$

where A_{mn} indicates the linking state of two nodes such as i and j, k_i is the count of the linked nodes for the node i and $\lambda(s_i, s_j)$ notes that whether the two nodes is belong to the same community.

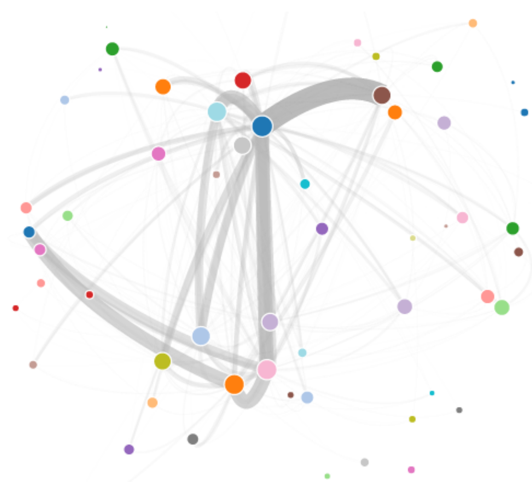


Figure 2: *Simplified community visualization.*

In order to elastically detect the communities, we remove the function of $\lambda(s_i, s_j)$ that requires complex calculation for a large graph. The optimized modularity Q' can be denoted as [Blondel et al.

2008]. Therefore we can define the hybrid modularity Q_h as follow:

$$Q_h = \begin{cases} Q, (c_v + c_l) \leq \varepsilon \\ Q', (c_v + c_l) > \varepsilon \end{cases} \quad (2)$$

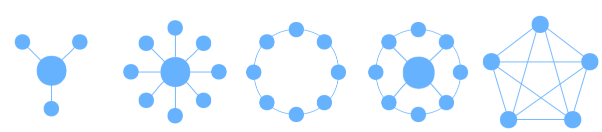
where ε means a parameter of the hybrid modularity, which is based on the power of computing. c_v indicates the vertex count of the graph and c_l indicates the edge count of the graph.

By using the hybrid modularity-based method, we can effectively detect the communities as shown in Fig. 1(left). Different communities are assigned with different colors. In order to observe the graph clearly, each community can be visualized as a node instead of crowd nodes as shown in Fig. 2. Modules are easy for the people to gain insights of the whole structure of the graph.

3.2 Module Pattern Definition

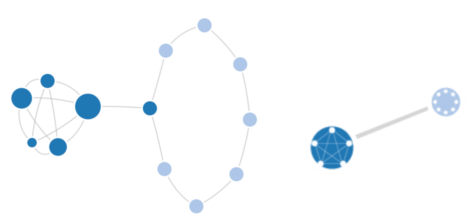
In order to further gain insights of modules, we analyze the patterns of each community. Wernicke [Wernicke 2006] present a method to detect motifs in large network. Motifs indicates the link patterns in the graph. Inspired from motifs, we calculate several pattern factors to analyze each module type of its internal graph, as well as other significant features such as module size, linking weight and visibility.

We design the module as five types such as low-center, high-center, circle, low-connection and high-connection as shown in Fig. 3. The types can be adjusted according to the requirement. The type of module can be detected from the link count statistics of each node in the internal graph.



(a) *l-center* **(b)** *h-center* **(c)** *circle* **(d)** *l-con* **(e)** *h-con*

Figure 3: *Five types of graph modules.*



(a) *Original graph* **(b)** *ModulGraph*

Figure 4: *A simple example of ModulGraph.*

3.3 Hierarchical Module

When the graph is massive, the community viewing may not be satisfied on the limited display. Based on our *Module Detection* method, we can extend it with hierarchical module method. Hierarchical module means that the detected communities in the previous detection process can be considered as the input of new community detection.

Furthermore, the definition of the module in the graph could be applied for the whole graph after hierarchically module the massive graph. We define the factor of screen size as w, h and the iteration of module as i. The default value of i is 1. If the count of the nodes in the graph is far greater than ηwh, we increase the module iteration of i until the size of node in new graph is nearly the value of ηwh. η is a free parameter that reflects the blank space among the nodes in the display.

3.4 Visualization Design

The basic *ModulGraph* layout can be calculated by using direct-force algorithm in order to ensure each module will not be overlapping with other ones. The basic idea of our visualization design for *ModulGraph* is try to visualize the specific patterns instead of the crowd nodes in a large graph. The visual elements of each module includes a circle with different radius called m_c, a bound with different width called m_b, an icon that indicates the module pattern called m_p and the a set of edges that connect the related modules called m_e. Each edge between modules has different width according to the weight of the edge.

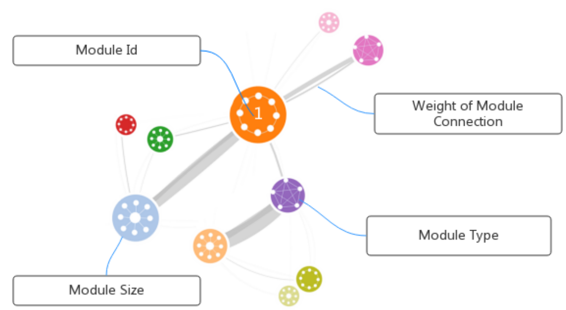


Figure 5: *A description of visualization design.*

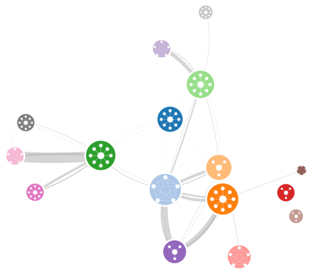
We choose the color themes from the Google themes to indicate different modules. Since the color is still limited, number in the center of the module can be shown to indicate the id of the module. When the modules are too many to display, it is available visualize the *ModulGraph* through a hierarchical way that is similar to the level of detail method [Zinsmaier et al. 2012].

4 Experiments and Results

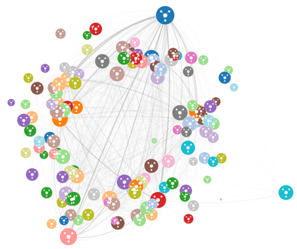
In the implementation, we use D3 [Bostock et al. 2011] to visualize the result on a computer with 2.8GHz Intel i7 CPU and 16GB RAM. We implement the visualizations in the Chrome browser. We select several large graph datasets from SNAP [Leskovec and Krevl 2014] that is a data library for analyzing of large information networks. The selected dataset is from Facebook, DBLP, YouTube, LiveJournal and Orkut as shown in Table. 1. Some results of ModulGraph are shown in Fig. 6. From the result we can denote that the graph patterns are easy to find.

Table 1: *The details of the selected datasets.*

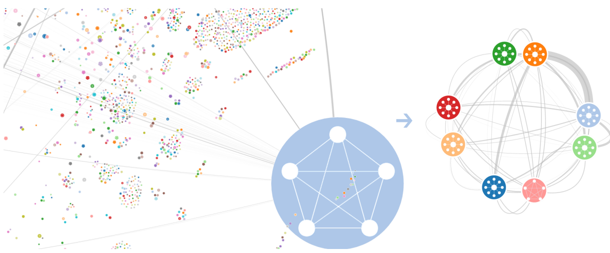
Datasets	Nodes	Edges	Module	Brief Modules
Facebook	4,039	88,234	16	-
DBLP	317,080	1,049,866	442	-
Youtube	1,134,890	2,987,624	9821	8
LiveJournal	3,997,962	34,681,189	21327	6
Orkut	3,072,441	117,185,083	1969	244



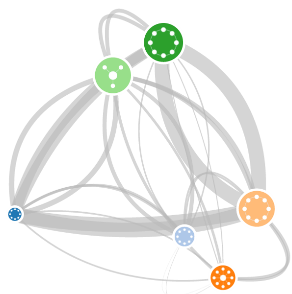
(a) *ModulGraph of Facebook social network.*



(b) *ModulGraph of DBLP network.*



(c) *ModulGraph of YouTube social network.*



(d) *The most simplified LiveJournal network.*

(e) *The most simplified Orkut network.*

Figure 6: *The results of ModulGraph.*

5 Conclusion

In this paper, we present the *ModulGraph*, a visualization method for visualizing large graphs. A hierarchical community detection method is presented to integrate the nodes that belong to the same community into a new integral module. Second, we give an available visual method to present the modules and show the relationships among them.

In the future, we plan to visualize graphs from streaming data as *dynamic ModulGraph*. The context change between the modules should be considered. Since the capacity of a single computer is limited for large graph computation, we plan to build on top of GraphX [Xin et al. 2013] and modify the modularity model to further achieve interactive visualization of massive graphs.

Acknowledgements

The authors would like to acknowledge the partial support of the Hong Kong Research Grants Council Grant GRF PolyU 5100/12E.

References

BAXTER, G., DOROGOVTSEV, S., GOLTSEV, A., AND MENDES, J. 2012. k-Core Organization in Complex Networks. In *Handbook of Optimization in Complex Networks*. 229–252.

BLONDEL, V. D., GUILLAUME, J.-L., LAMBIOTTE, R., AND LEFEBVRE, E. 2008. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment 2008*, 10, P10008.

BOSTOCK, M., OGIEVETSKY, V., AND HEER, J. 2011. D3: Data-Driven Documents. *IEEE Transactions on Visualization and Computer Graphics 17*, 12, 2301–2309.

CHAE, S., MAJUMDER, A., AND GOPI, M. 2012. HD-GraphViz: Highly Distributed Graph Visualization on Tiled Displays. In *Proceedings of the Eighth Indian Conference on Computer Vision, Graphics and Image Processing*, ICVGIP '12, 43:1—-43:8.

DUNNE, C., AND SHNEIDERMAN, B. 2013. Motif simplification: Improving network visualization readability with fan, connector, and clique glyphs. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, CHI '13, 3247–3256.

HORNG, D., CHAU, P., KITTUR, A., HONG, J. I., AND FALOUTSOS, C. 2011. Apolo : Making Sense of Large Network Data by Combining Rich User Interaction and Machine Learning. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 167–176.

LESKOVEC, J., AND KREVL, A., 2014. SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`, June.

NEWMAN, M. E. J. 2006. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences 103*, 23, 8577–8582.

NEWMAN, M. E. J. 2011. Communities, modules and large-scale structure in networks. *Nature Physics 8*, 1, 25–31.

ROSVALL, M., AND BERGSTROM, C. T. 2008. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences 105*, 4, 1118–1123.

SHI, L., CAO, N., LIU, S., QIAN, W., TAN, L., WANG, G., SUN, J., AND LIN, C.-Y. 2009. Himap: Adaptive visualization of large-scale online social networks. In *Visualization Symposium, 2009. PacificVis' 09. IEEE Pacific*, IEEE, 41–48.

TIAN, Y., HANKINS, R. A., AND PATEL, J. M. 2008. Efficient Aggregation for Graph Summarization. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, 567–580.

WERNICKE, S. 2006. Efficient detection of network motifs. *IEEE/ACM Transactions on Computational Biology and Bioinformatics 3*, 4, 347–359.

XIN, R. S., GONZALEZ, J. E., FRANKLIN, M. J., STOICA, I., AND AMPLAB, E. 2013. GraphX: A Resilient Distributed Graph System on Spark. *First International Workshop on Graph Data Management Experiences and Systems*, 2:1—-2:6.

ZINSMAIER, M., BRANDES, U., DEUSSEN, O., AND STROBELT, H. 2012. Interactive Level-of-Detail Rendering of Large Graphs. *IEEE Transactions on Visualization and Computer Graphics 18*, 12, 2486–2495.