A Comparative Analysis of Large-scale Network Visualization Tools

S M Arifuzzaman

2018 IEEE International Conference on Big Data (Big Data)

A Comparative Analysis of Large-scale Network Visualization Tools

Md Abdul Motaleb Faysal and Shaikh Arifuzzaman
Department of Computer Science, University of New Orleans
New Orleans, LA 70148, USA.
Email: mfaysal@my.uno.edu, smarifuz@uno.edu

***Abstract*—Network (Graph) is a powerful abstraction for representing underlying relations and structures in large complex systems. Network visualization provides a convenient way to explore and study such structures and reveal useful insights. There exist several network visualization tools; however, these vary in terms of scalability, analytics feature, and user-friendliness. Due to the huge growth of social, biological, and other scientific data, the corresponding network data is also large. Visualizing such large network poses another level of difficulty. In this paper, we identify several popular network visualization tools and provide a comparative analysis based on the features and operations these tools support. We demonstrate empirically how those tools scale to large networks. We also provide several case studies of visual analytics on large network data and assess performances of the tools. We show both runtime and memory efficiency of the tools while using layout algorithms and other network analysis methods.**

***Index Terms*—Big networks; Visualization; Visual analytics; Network analytics; Graph mining; Scalable algorithms**

I. INTRODUCTION

Network visualization is an important part of graph-based data analysis and research. Networks are often the most convenient way to represent interactions among entities in social, biological, infrastructure and other information systems [7], [8]. Examples include interaction among persons in social networks, connectivity of web pages in world wide web graphs, and interaction of proteins in biological networks [29]. Network analysis and visualization help discover structures and patterns in a network and thereby reveal useful insights [8].

Large-scale network visualization has been a field of research interest for more than a few years [3], [1], [5]. There exists a number of tools that offer different functionalities for visualizing, analyzing, and filtering networks [32], [33], [34], [30]. Many of these tools are developed considering a particular application domain such as biological data [1] and social networks [4]. Developer packages are available in programming languages such as python and R to display network data. Some tools provide visualization along with a suite of network algorithms [1], [2]. Some of the tools [30], [31] require programming background while others [37], [39], [42] do not. Some tools provide web-interface [34], [35] while some others [1], [32] run on desktop. Further, the emergence of large network data necessitates the development of scalable tools [5], [6]. The existing tools show a varying degree of scalability. Therefore, there is a need for studying existing network visualization tools based on the general usability, user-friendliness, use case scenarios about how those tools and which of the tools can be used for a specific network data, and scalability for large network analysis. Such study will help a person to decide which tool to use for a specific case based on his need.

One such study in [3] compares a couple of tools based on scalability. Comparative studies on other visualization metrics should be conducted to let end users have freedom in choosing the specific tool he needs to use. In this paper, we have chosen five widely used visualization tools for an empirical and comparative analysis. Our comparisons are based on factors such as supported file formats, scalability in visualization and analysis, interacting capability with the drawn network, end user-friendliness (e.g., users with no programming background, cost-effectiveness), and developer friendliness (open-source, multi-platform supports) and support for useful visual analytics.

In this paper, we focused mainly on the network visualization tools that are free, open-source, desktop-based and do not require programming background to perform operations and discover insights from a network. Table I lists the five tools we study along with their versions we used and compatible operating systems. We also provide a description of some other prominent visualization tools in Table II.

Next, we present the network visualization tools we chose, discuss the features and components, provide some case studies of visual analytics on real-world scenarios, and conduct comparative analysis on the functional capabilities.

II. TOOLS FOR LARGE-SCALE VISUAL ANALYTICS ON NETWORKS

We provide a description of five popular network visualization tools below.

A. Cytoscape

Cytoscape [1] is an open source software platform for integrating, visualizing, and analyzing network data. The tool originally developed for processing biomolecular interaction networks [16] can also be used for general complex network analysis.

Features. Cytoscape is a powerful tool for visualization and analyzing protein-protein interaction (PPI) data in biological network. This tool supports a number of layouts to view the graph data in a convenient way, e.g., hierarchical layout, Circular layout, orthogonal layout, and tree layout. One powerful feature of Cytoscape is its App Store from where a number of applications can be downloaded. Each of those applications comes with different graph analytics and visualization features that serve different purposes such as creating layouts, computing graph metrics, and clustering.

TABLE I
DESCRIPTION OF THE GRAPH VISUALIZATION TOOLS

Name	Version	Tool Type	Development Core	Supported Platforms
Cytoscape [1]	3.6.1	Free, Open Source	Java	Linux, Mac, Windows
Gephi [2]	0.9.2	Free, Open Source	Java, OpenGL	Linux, Mac, Windows
Pajek [5]	5.05	Free	Delphi (Pascal)	Windows
SocNetV [4]	2.4	Free, Open Source	C++/Qt	Linux, Mac, Windows
Tulip [6]	5.1.0	Free, Open Source	C++	Linux, Mac, Windows

TABLE II
LISTING OF SOME PROMINENT NETWORK VISUALIZATION AND ANALYTICS TOOLS

Tool Name	Description
Alchemy.js [30]	A web based graph visualization application for developers in JavaScript with capability to cluster, filter and embed network.
Arbor.js [31]	A web based force-directed graph visualization library for developers.
Circos [32]	Initially designed for displaying genomic data, this desktop based tool can be used to visualize graph in circular layout.
Grano [33]	An open-source tool to track networks of political or economic interest. It helps to merge data from different sources.
Kumu [34]	A web-based data visualization platform that can organize complex information into interactive relationship maps. Commerical tool.
Polinode [35]	A cloud based graph visualization and analysis service. It's a commercial tool.
NodeXL [36]	A desktop based tool and Excel plugin. The commercial version comes with a good interface for social data collection and analysis.
Graphviz [37]	An open source software where the layout programs take descriptions of graphs in text language and make diagrams to be exported as image, pdf, or to be displayed in interactive graph browser.
Sigma js [38]	A JavaScript library to draw network and specifically helpful for displaying network in web pages.
Textexture [39]	With this web-based application any text can be represented as a graph and thus textual data can be visualized as a network.
Ucinet [11]	Ucinet is widely used in academic research. This commercial tool is packed with a number of graph based metrics analytics, visualizations of the outcome of those analytics have scope for improvement.
NetworkX [40]	Python programming software package for creation and manipulation of network. Can process a large number of vertices and is open source.
igraph [41]	For graph analytics for developers with experience in either C, python or R, this is an open source package/library.
VIS [42]	VIS stands for visual investigative scenarios. As the name suggests, this is a web based application to assist investigators, journalists in mapping business or crime networks.

B. Gephi

Gephi [2] is known as a software for both visualization and performing graph-based analysis on network. The tool also allows users to interact with the drawn network through on-screen tools in the visualization window.

Features. Gephi is a great tool for visualization of a network. The tool can display a network of 875,713 vertices and 5,105,039 edges (web-Google network from SNAP datasets [29]). The feature to load and operate on large network is significant. Gephi can calculate different centrality measures (closeness, betweenness, eccentricity), perform graph-based operations such as finding graph density, avg. weighted degree, page rank, and connected components, also perform node-based operations, e.g., finding avg clustering coefficient, edge-based operations, e.g., finding average path length. The tool can display a graph in a number of different layouts such as Yifan Hu, Fruchterman Reingold [12], rotation, OpenOrd [13], Force Atlas, and Noverlap. The tool can be used to detect community in a network using Louvain [14] algorithm. This tool has good styling features like changing the appearance, size, color of vertices and edges, which help to visualize the outcome of an operation efficiently.

C. Pajek

Pajek [5] provides a desktop-based platform for visualization and analysis on network data. It has both commercial and non-commercial versions.

Features. Pajek can draw a network with millions of nodes. Pajek has three versions based on the network size– Pajek, PajekXXL and Pajek3XL. In our study, we used the version named Pajek. Network visualization in Pajek is not on per with Gephi but when it comes to speed of operation on large datasets, Pajek can surpass most of the graph analytics tools available. Pajek supports layouts such as Fruchterman-Reingold, Kamada-Kawai [21], circular, Pivot Multidimensional Scaling, and Visualization of Similarity(VOS) based mapping. It can apply style features like coloring, sizing of vertices and edges in the drawing window. Pajek supports calculation of different centrality, degree, coefficient metrics, discovery of cluster and communities, transformations over a network.

D. SocNetV

SocNetV [4] is another desktop based tool, which was primarily developed for visualization and analysis of social network.

Features. SocNetV can run community detection algorithm, clique discovery, and triad census [16]. It can calculate closeness centrality, betweenness centrality, degree centrality. The tool supports the calculation of standard graph and network cohesion metrics, e.g., distances, connectedness, density, and clustering coefficients. It has web crawler to create interaction network from the URL provided.

E. Tulip

Tulip [6] is a network visualization and analysis software developed by David Auber et al. from the University of

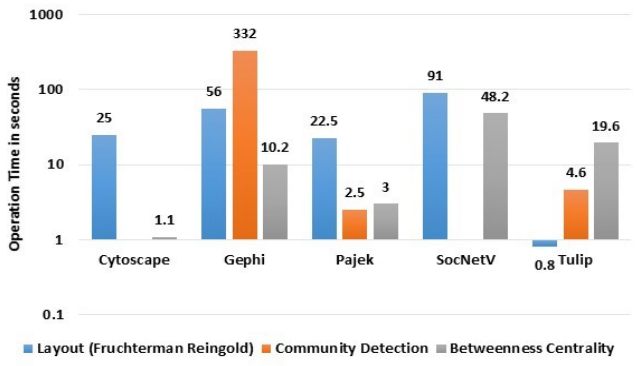

Fig. 1. Comparison of runtime for various operations performed by the tools

Bordeaux, France. This software is packed with a large set of graph-based operations.

Features. Tulip draws graphs using layouts such as circular or random, can perform topological tests whether the graph is connected, biconnected, acyclic, and planer. The tool can also perform basic graph-based operations such as spanning tree, and topological update. Tulip can draw Delaunay triangulation and Voronoi diagram of a given network. It can calculate betweenness centrality, page rank, eccentricity, and Welsh Powell for graph coloring. There are clustering and community detection functionalities in this tool as well. It can perform Louvain community detection algorithm, Markov Clustering Algorithm, and Link Communities. This makes this tool a great option for running clustering operations.

III. Experimental Evaluation: A Comparative Analysis

We present a comparative analysis based on different features and capability of our selected visualization tools. The experiments were conducted in a 64 bit machine with 16 GB RAM, Intel(R) Core(TM) i7-4770 CPU with 3.40GHz clock speed, Windows 10 operating system.

A. Comparison of Runtime Performance

We compare the runtime performance of our selected tools by running layout algorithm, community detection, and prominence measure (Betweenness Centrality). Fig.1 shows the runtime comparison. The specific graph layout used is Fruchterman Reingold which is a force directed layout. All of the five tools have these layout display capability. The layout was applied on a network with 1285 vertices and 7524 edges (EuroSiS web mapping study network [27]). For community detection, we used the Louvain [14] method. Three of the tools, Gephi, Pajek, and Tulip have this community detection algorithm. The other two tools Cytoscape and SocNetV do not have Louvain method and hence are omitted. The network on which we ran Louvain is email-Enron from SNAP datasets [29], which have 36692 vertices and 367662 edges. For calculating Betweenness Centrality we used the Power grid network [29] which is an undirected, unweighted network with 4941 vertices and 6594 edges representing the topology of the Western States Power Grid of the United States. As shown in Fig. 1, Tulip renders the graph layout faster than others. Tulip and Pajek also have fast implementation of Louvain method of community detection. Pajek and Cytoscape compute the betweenness centrality values faster than the other tools.

B. Scalability to Large Networks

In Table III, we present the memory consumption in megabytes and reading and displaying time in seconds by our selected tools for several networks from SNAP datasets [29]. The empty cells indicate inability to both read and display the network. SocNetV is not listed here as it was not able to read and display any of the networks in the table. As shown in the table, Gephi is able to read and visualize all these networks. Cytoscape supports visualization of networks of size up to a couple of million edges. The visualization with Pajek and Tulip do not scale to very large network data.

C. Support for a Diverse File Formats

Pajek only supports networks in Pajek(.net) format. Cytoscape supports GML(.gml), GraphML(.graphml), NNF(.nnf), and SIF(.sif) formats. SocNetV supports Edge list(.txt), UCINET(.dl), Pajek(.net), and GML(.gml) formats. Although it can read a network in GML format but cannot display the edges. SocNetV crashes when it tries to display the Graphviz(.dot) format. Tulip can display networks stored as UCINET(.dl), Pajek(.net), GML(.gml), GraphML(.graphml), TLP(.tlp), GEXF(.gexf), Graphviz(.dot), and BibTeX (.bib) formats. Gephi supports a number of file formats for graph. Supported formats include GEPHI(.gephi), UCINET(.dl), Pajek(.net), GML(.gml), GraphML(.graphml), GEXF(.gexf), Graphviz(.dot), and GDF(.gdf) [15].

D. Reports/Feedback to Users

Gephi has a convenient reporting mechanism after running an operation. For instance, after running any statistical operation Gephi displays an HTML report with the options to print, copy or save the report. SocNet has similar reporting strategy. Pajek provides a log of the executed operations. Cytoscape and Tulip do not have such kind of reporting strategy although the resultant graph can be saved as a result of the operation in specified formats.

E. Layout Methods for Displaying Networks

Gephi is packed with many graph rendering algorithms convenient for visualizing thousands or more vertices. For instance, Fruchterman Reingold algorithm [12], also known as spring-electrical model, is a force directed layout drawing algorithm, where a combination of two forces applied between two vertices. The repulsive force is inversely proportional to the distance between them, whereas the attractive force exists only between neighboring vertices and is proportional to the square of the distance. Fig. 2 shows a drawing for netscience [17] graph with Fruchterman Reigold algorithm. Gephi also includes layout modeling algorithms such as Yifan Hu, OpenOrd [18], and Random Layout. When it comes to graph display

TABLE III
CAPABILITY FOR VISUALIZATION OF LARGE NETWORKS: SCALABILITY IN MEMORY AND TIME

Dataset	# Vertices	# Edges	Cytoscape		Gephi		Pajek		Tulip	
			Mem(MB)	Time(sec)	Mem(MB)	Time(sec)	Mem(MB)	Time(sec)	Mem(MB)	Time(sec)
No Graph	N/A	N/A	960	N/A	288	N/A	60	N/A	70	N/A
email-Enron	36692	367662	3200	14	1100	13	88	3	658	434
web-Stanford	281903	2312497	12300	125	3970	101	-	-	-	-
web-Google	875713	5105039	-	-	6350	280	-	-	-	-
wiki-Talk	2394385	5021410	-	-	7742	612	-	-	-	-

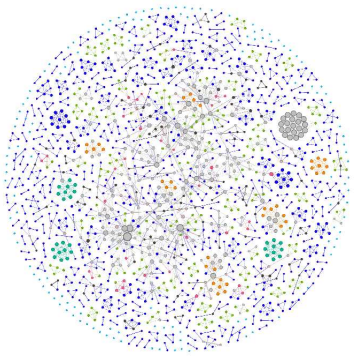

Fig. 2. Fruchterman Reingold layout drawn by Gephi.

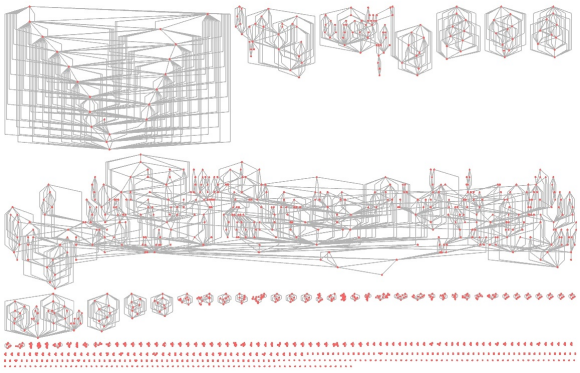

Fig. 3. Graph drawn in Tulip using Sugiyama [22] layout.

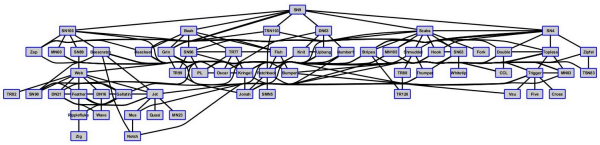

Fig. 4. A tree Layout drawn by Cytoscape tool.

layout, Tulip is packed with lots of algorithms which are categorized by different classes. It has implementation of many force-directed layout algorithms such as Fruchterman Reingold, GEM (short for Graph Embedder), GRIP [19], Kamada Kawai [21]. Tree based layouts include Bubble Tree, Cone Tree, Tree Radial, and Ortho Tree. Hierarchical Layouts include Sugiyama [22], and Balloon. Cytoscape has layout plugins to draw and manipulate graph in different ways. One such plugin is *yFiles* which comes with layout modeling algorithms, e.g., Circular, Hierarchical, Organic, Orthogonal, and tree. Plugins can be installed on demand from Cytoscape App Store to draw graphs in different layouts. Pajek can apply force-directed layouts, e.g., Fruchterman-Reingold, Kamada-Kawai, with the options to change parameters. Pajek has some other graph layout algorithms based on multidimensional scaling (MDS), and VOS [23]. SocNetV has similar force-directed layout algorithms, e.g., Fruchterman-Reingold, Kamada-Kawai and Eades Spring Embedder [24].

F. Interacting with Individual Vertex in a network

We examine how interactive a visualization is when drawn by each tool. Such interaction is critical in visual analytics, especially for identifying prominent nodes in the network and exploring its neighborhood. For Gephi (Fig.5) and Tulip (Fig.6), we can highlight the neighborhood of a vertex by selecting that vertex in the graphical window. With Cytoscape, one can select a vertex from the graphical window and then find the neighbors of that vertex (see Fig.7). Further, the neighbors are listed in the Table Panel. For Pajek and SocNetV, the feature of highlighting the neighbors of a selected node in Graphical window does not exist. In Pajek we can select a group of vertices by using a selection area using the mouse right button, which effectively results in zooming over of that region. In tulip, we can also search an individual vertex by its label or its properties using the query section of the Elements window. With Cytoscape and SocNetV, one can look up a vertex by textual input in the graphical window, i.e., by vertex label or vertex id, which returns the highlighted vertex along with its neighborhood.

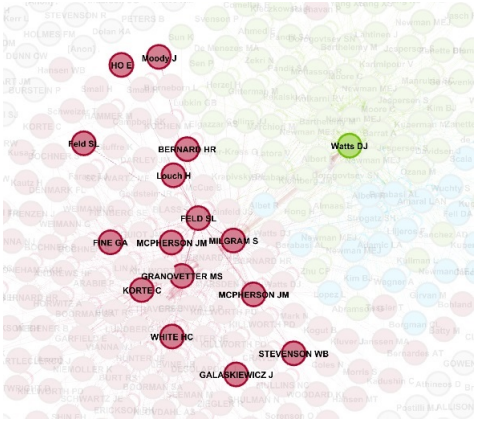

Fig. 5. Highlighting neighborhood of a vertex with Gephi.

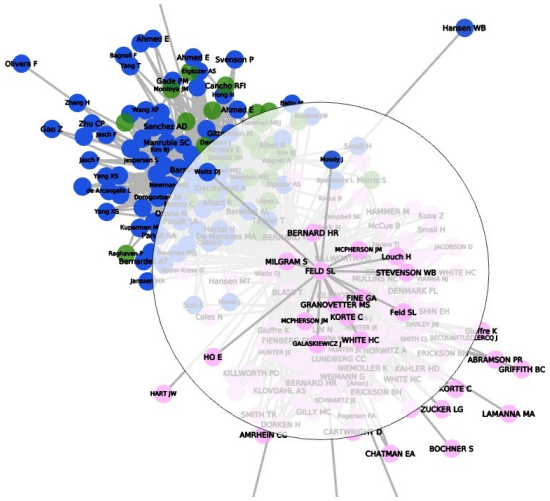

Fig. 6. Highlighting neighborhood with Tulip.

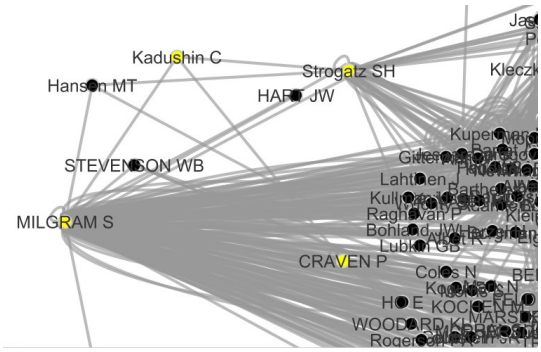

Fig. 7. Highlighting neighborhood with Cytoscape.

G. Quality of Visualization: Vividness and Sharpness

Gephi and Tulip stand out in quality of generated visualization. They render high quality drawing of a graph with pleasing view to the eyes. Additionally, the resulted visualizations fit in the display screen nicely. Networks drawn with SocNetV also fit well on display window. Visualizations drawn by Cytoscape is less vivid compared to others and sometimes do not fit inside the display screen. Visualization in Pajek is sharp; however, sometimes one might experience images not fitting in the screen.

IV. Applications: Visual Analytics on Large Networks

We present several case studies of visual analytics in real-world scenarios.

A. Identifying Bridge Nodes

In a network, there exist nodes (vertices) that can connect multiple separate groups of nodes. We call them making 'bridges' among groups. Such bridge nodes play important roles in the corresponding systems. For example, those few people in contact networks are critical in spreading information (or rumors, disease, etc.) among the groups. In an infrastructure network, these nodes pose communication vulnerability (see bridge/articulation point in [29]). To visualize bridge nodes of a network by our selected tools, we used US Air flights 1997 [26] network as a sample network, where the data represents the flights between North American airports on the year of 1997. We ran betweenness centrality measure to find out the airports in this region that act as gateway hub or bridge node during that time. The airports with high betweenness centralities are the ones used heavily for connecting flights. It also means those airports were the busiest airports where any functional disruption would affect many flights. We used the tools to find the betweenness centrality of the network and found such airports of critical importance. All of the five tools identified the following airports as bridge nodes: Chicago O'hare Intl., Anchorage Intl., Dallas/Fort Worth Intl., San Francisco Intl., Seattle-Tacoma Intl., Lambert-St Louis Intl. and The William B Hartsfield Atlanta. See the visualization in Fig. 8 generated by Tulip.

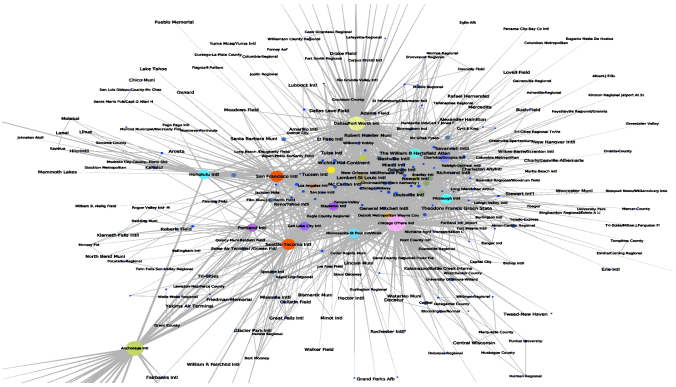

Fig. 8. Bridge nodes (bigger in picture) of North American Flights 1997

B. Visualizing communities in a network

Discovering communities in a network has significant application in academia, business, marketing, social networking, and criminal investigations. For this analytics, we use a citation network [25] of 396 vertices and 1555 edges. We use Louvain [14] community detection algorithm available with three (Gephi, Pajek, Tulip) of our five tools.

We find that there are three large communities detected by each of the tools. Group of vertices marked by Gephi as red, light green and blue belongs to the same communities in Fig.9. Group of vertices marked as pink, deep blue and green in belongs to the same communities respectively as identified by Tulip in Fig.10. Pajek also generates similar visualization with three major communities.

V. Conclusion

Network Visualization and analytics are critical to gaining useful insights from big network data. In this paper, we present a comparative analysis of popular network visualization tools by considering the factors that matter most during both visualization and performing analytics. All of the selected tools demonstrate the ability to display and analyze large

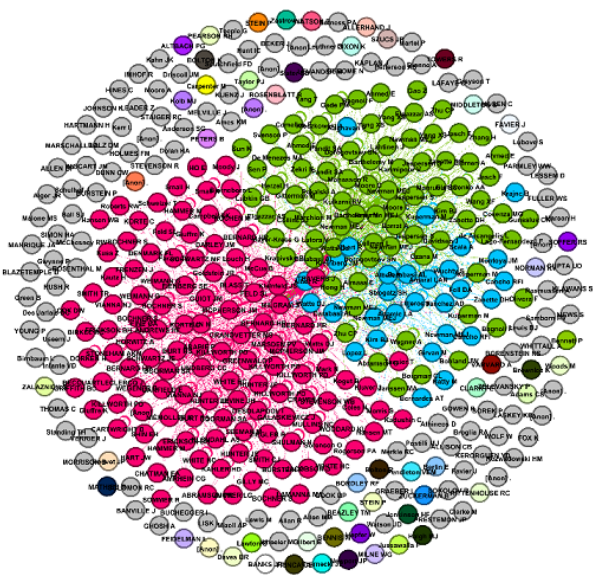

Fig. 9. Communities detected in Gephi from citation network

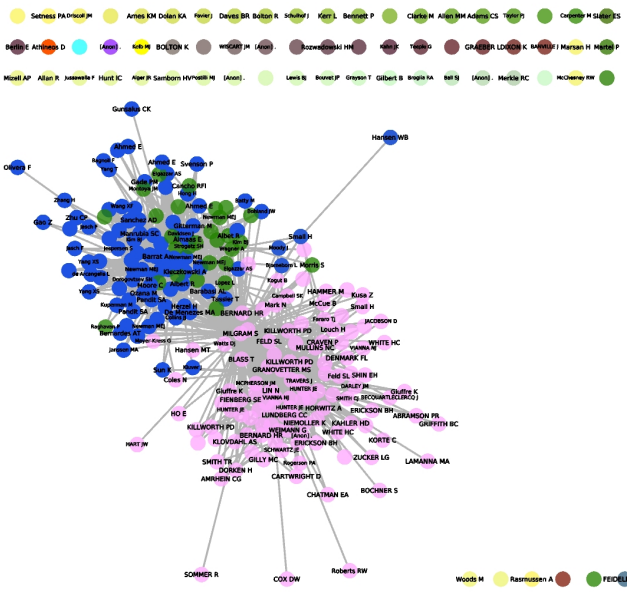

Fig. 10. Communities detected in Tulip from citation network

networks with strengths of varying degrees of scalability and applicability. We believe our study will serve as a useful guide to network scientists or researchers in selecting an appropriate network visualization tool based on one's specific need.

REFERENCES

[1] Shannon P, Markiel A, Ozier O, Baliga NS, Wang JT, Ramage D, Amin N, Schwikowski B, Ideker T. Cytoscape: a software environment for integrated models of biomolecular interaction networks, Genome Research 2003 Nov; 13(11):2498-504

[2] Mathieu Bastian and Sebastien Heymann and Mathieu Jacomy, Gephi: An Open Source Software for Exploring and Manipulating Networks, International AAAI Conference on Weblogs and Social Media, 2009

[3] Georgios A. Pavlopoulos, David Paez-Espino, Nikos C. Kyrpides and Ioannis Iliopoulos, Empirical Comparison of Visualization Tools for Larger-Scale Network Analysis, Advances in Bioinformatics, Volume 2017, Article ID 1278932

[4] Social Network Visualizer, http://socnetv.org/. [Accessed: 11/19/2018]

[5] Vladimir Batagelj and Andrej Mrvar, PajekProgram for Large Network Analysis

[6] David Auber, Tulip A Huge Graph Visualization Framework, Graph Drawing Software, Springer Berlin Heidelberg, 2004, p 105–126

[7] R. A. Becker, S. G. Eick and A. R. Wilks, "Visualizing network data," in IEEE Transactions on Visualization and Computer Graphics, vol. 1, no. 1, pp. 16-28, March 1995.

[8] Ivan Herman, Guy Melancon, and M. Scott Marshall, Graph Visualization and Navigation in Information Visualization: A Survey, IEEE Transactions on Visualization and Computer Graphics, Vol. 6, No. 1, January-March 2000

[9] EuroSiS, https://github.com/gephi/gephi/wiki/Datasets

[10] http://dataviz.tools/category/network-visualization/

[11] UCINET, https://sites.google.com/site/ucinetsoftware/home

[12] T. Fruchterman and E. Reingold. Graph drawing by force-directed placement. Softw. Pract. Exp., 21(11):11291164, 1991

[13] Martin, Shawn and Brown, W Michael and Klavans, Richard and Boyack, Kevin W, OpenOrd: an open-source toolbox for large graph layout, Visualization and Data Analysis 2011, volume 7868, pages 786806, International Society for Optics and Photonics

[14] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, Etienne Lefebvre, Fast unfolding of communities in large networks, Journal of Statistical Mechanics: Theory and Experiment 2008 (10), P10008 (12pp)

[15] Geographic Data Files, http://changelog.ca/log/2013/03/09/gdf

[16] Paul W. Holland, Samuel Leinhardt, A Method for Detecting Structure in Sociometric Data, November 1970, American Journal of Sociology 73(3):492-513

[17] M.E.J. Newman, Finding community structure in networks using the eigenvectors of matrices, Preprint physics/0605087 (2006)

[18] S. Martin, W. M. Brown, R. Klavans, and K. Boyack (to appear, 2011), "OpenOrd: An Open-Source Toolbox for Large Graph Layout," SPIE Conference on Visualization and Data Analysis (VDA)

[19] Pawel Gajer and Stephen G. Kobourov, GRIP: Graph Drawing with Intelligent Placement, 8th Symposium on Graph Drawing (GD), p. 222-228, 2000

[20] Arne Frick, Andreas Ludwig and Heiko Mehldau, A Fast Adaptive Layout Algorithm for Undirected Graphs (Extended Abstract and System Demonstration)

[21] Tomihisa Kamada and Satoru Kawai, An algorithm for drawing general undirected graphs, Information Processing Letters Volume 31, Issue 1, 12 April 1989, Pages 7-15

[22] Kozo Sugiyama and Kazuo Misue, Graph Drawing by the Magnetic Spring Model, Journal of Visual Languages & Computing Volume 6, Issue 3, September 1995, Pages 217-231

[23] Nees Jan van Eck, Ludo Waltman, Rommert Dekker and Jan van den Berg, A Comparison of Two Techniques for Bibliometric Mapping: Multidimensional Scaling and VOS

[24] P. Eades. A heuristic for graph drawing. Congressus Numerantium, 42:149160, 1984

[25] citation network, http://www.cs.umd.edu/ golbeck/INST633o/Viz.shtml

[26] North American Transportation Atlas Data (NORTAD), http://vlado.fmf.uni-lj.si/pub/networks/data/map/USAir97.net

[27] EuroSiS web mapping study, https://github.com/gephi/gephi/wiki/Datasets

[28] D. J. Watts and S. H. Strogatz, Nature 393, 440-442 (1998)

[29] Jure Leskovec and Andrej Krevl, SNAP Datasets: Stanford Large Network Dataset Collection, http://snap.stanford.edu/data, June-2014

[30] Alchemy.js, http://graphalchemist.github.io/Alchemy

[31] arbor.js, http://arborjs.org [Accessed: 11/19/2018]

[32] Circos, http://www.circos.ca [Accessed: 11/19/2018]

[33] GRANO, http://granoproject.org [Accessed: 11/19/2018]

[34] Kumu, https://kumu.io [Accessed: 11/19/2018]

[35] polinode, https://www.polinode.com [Accessed: 11/19/2018]

[36] NodeXL, https://www.smrfoundation.org [Accessed: 11/19/2018]

[37] Graphviz, http://www.graphviz.org [Accessed: 11/19/2018]

[38] Sigma js, http://sigmajs.org [Accessed: 11/19/2018]

[39] textexture, http://textexture.com/index.php [Accessed: 11/19/2018]

[40] NetworkX, https://networkx.github.io [Accessed: 11/19/2018]

[41] igraph, http://igraph.org/redirect.html [Accessed: 11/19/2018]

[42] Visual Investigative Scenarios (VIS), https://vis.occrp.org