Hindawi
Advances in Bioinformatics
Volume 2017, Article ID 1278932, 8 pages
https://doi.org/10.1155/2017/1278932

Review Article

Empirical Comparison of Visualization Tools for Larger-Scale Network Analysis

Georgios A. Pavlopoulos,[1] David Paez-Espino,[1] Nikos C. Kyrpides,[1] and Ioannis Iliopoulos[2]

[1]*Department of Energy, Joint Genome Institute, Lawrence Berkeley Labs, 2800 Mitchell Drive, Walnut Creek, CA 94598, USA*
[2]*Division of Basic Sciences, University of Crete Medical School, Andrea Kalokerinou Street, Heraklion, Greece*

Correspondence should be addressed to Georgios A. Pavlopoulos; g.pavlopoulos@gmail.com and Ioannis Iliopoulos; iliopj@med.uoc.gr

Received 22 February 2017; Revised 14 May 2017; Accepted 4 June 2017; Published 18 July 2017

Academic Editor: Klaus Jung

Gene expression, signal transduction, protein/chemical interactions, biomedical literature cooccurrences, and other concepts are often captured in biological network representations where nodes represent a certain bioentity and edges the connections between them. While many tools to manipulate, visualize, and interactively explore such networks already exist, only few of them can scale up and follow today's indisputable information growth. In this review, we shortly list a catalog of available network visualization tools and, from a user-experience point of view, we identify four candidate tools suitable for larger-scale network analysis, visualization, and exploration. We comment on their strengths and their weaknesses and empirically discuss their scalability, user friendliness, and postvisualization capabilities.

1. Background

Health and natural sciences have become protagonists in the big-data world as high-throughput advances continuously contribute to the exponential growth of data volumes. Nowadays, biological repositories expand every day by hosting various entities such as proteins, genes, drugs, chemicals, ontologies, functions, articles, and the interactions between them, often leading to large-scale networks of thousands or even millions of nodes and connections. As such networks are characterized by different properties and topologies, graph theory comes to play a very important role by providing ways to efficiently store, analyze, and subsequently visualize them [1–5].

Visualization and exploration of biological networks at such scale are a computationally challenging task and many efforts in this direction have failed over the years. Recent review articles [3, 4, 6] discuss the challenges in the biological data visualization field and list a catalog of standalone and web-based visualization tools as well as the visual concepts they are implemented to serve. While these resources are valuable to capture the big picture in the field, get a sense of the available tools, and spot the strengths and the weaknesses of a tool of interest at a glance, no empirical feedback on the tools' scalability was obvious.

To shortly mention representative tools in the field, 2D standalone applications like graphVizdb [7], Ondex [8], Proviz [9], VizANT [10], GUESS [11], UCINET [12], MAPMAN [13], PATIKA [14], Medusa [15], or Osprey [16] as well as 3D visualization tools such as Arena3D [17, 18] and BioLayout Express [19] already exist. Each of them is designed to serve a different purpose. For example, Ondex is implemented to gather and manage data from diverse and heterogeneous datasets, Proviz is dedicated to handle protein-protein interaction datasets, VizANT focuses on metabolic networks and ecosystems, Medusa is able to show semantic networks and multiedged connections, GUESS supports dynamic and time sensitive data, Osprey is implemented to annotate biological networks, Arena3D is targeting multilayered graphs, and BioLayout Express is designed for generic advanced 3D network visualizations.

Despite the fact that such tools are widely used and have great potential for further development, to our experience, they are not recommended for large-scale network analysis in their current versions. UCINET windows application could potentially be used for just visualization purposes. Its absolute maximum network size is about 2 million nodes but, in practice, most of its procedures are too slow to run networks larger than about 5,000 nodes.

Among several existing tools that we tested, we find Cytoscape (v3.5.1) [20], Tulip (v4.10.0) [21], Gephi (v0.9.1) [22], and Pajek (v5.01) [23, 24] standalone applications to be the top four candidates for visualization, manipulation, exploration, and analysis of very big networks. For these four tools, we empirically evaluate their pros and cons, we comment on their scalability, user friendliness, layout speed, offered analyses, profiling, memory efficiency, and visual styles, and we provide tips and advice on which of their features can scale and which of them is better to avoid.

In order to show a representative visualization generated by these four tools, we constructed a graph consisting of 202,424 nodes and 354,468 edges showing the habitat distribution of 202,417 protein families across 7 habitats. Data was collected from the IMG integrated genome and metagenome comparative data analysis system [25] whereas protein families originate from public metagenomes only.

A step-by-step protocol describing how these images were generated is presented as Supplementary Material, available online at https://doi.org/10.1155/2017/1278932. Comments on problems which occurred during our analysis as well as drawbacks and strengths of the visualization tools used for the purposes of this review are extensively discussed.

2. Top Four Candidates for Large-Scale Network Visualization

2.1. Gephi (Version 0.9.1). Gephi is free open-source, leading visualization and exploration software for all kinds of networks and runs on Windows, Mac OS X, and Linux. It is our top preference as it is highly interactive and users can easily edit the node/edge shapes and colors to reveal hidden patterns. The aim of the tools is to assist users in pattern discovery and hypothesis making through efficient dynamic filtering and iterative visualization routines. As a generic tool, it is applicable to exploratory data analysis, link analysis, social network analysis, biological network analysis, and poster creation.

2.1.1. Scalability. Gephi comes with a very fast rendering engine and sophisticated data structures for object handling, thus making it one of the most suitable tools for large-scale network visualization. It offers very highly appealing visualizations and, in a typical computer, it can easily render networks up to 300,000 nodes and 1,000,000 edges. Compared to other tools, it comes with a very efficient multithreading scheme, and thus users can perform multiple analyses simultaneously without suffering from panel "freezing" issues.

2.1.2. Layouts. In large-scale network analysis, fast layout is a bottleneck as most sophisticated layout algorithms become CPU and memory greedy by requiring long running time to be completed. While Gephi comes with a great variety of layout algorithms, OpenOrd [26] and Yifan-Hu [27] force-directed algorithms are mostly recommended for large-scale network visualization. OpenOrd, for example, can scale up to over a million nodes in less than half an hour while Yifan-Hu is an ideal option to apply after the OpenOrd layout. Notably, Yifan-Hu layout can give aesthetically comparable views to the ones produced by the widely used but conservative and time-consuming Fruchterman and Reingold [28]. Other algorithms offered by Gephi are the circular, contraction, dual circle, random, MDS, Geo, Isometric, GraphViz, and Force atlas layouts. While most of them can run in an affordable running time, the combination of OpenOrd and Yifan-Hu seems to give the most appealing visualizations. Descent visualization is also offered by OpenOrd layout algorithm if a user stops the process when ~50–60% of the progress has been completed. Of course, efficient parametrization of any chosen layout algorithm will affect both the running time and the visual result.

2.1.3. Postvisualization Analysis. Edge-bundling and famous clustering algorithms such as the MCL [29] do not come by default with Gephi but can be downloaded from Gephi's plugin library (~100 plugins). In addition, GeoLayout Gephi's plugin is very suitable to plot a network with geographical information. Coming to dynamic network visualization, Gephi is the forefront of innovation with dynamic graph analysis. Users can visualize how a network evolves over time by manipulating its embedded timeline. While visualization of a network over time is something very useful, its current algorithms are not suitable for large-scale networks. Similarly, for large-scale networks it is highly recommended for users to apply clustering algorithms using external command line applications and then import the clustering results to a visualization tool.

To study a network's topology, Gephi comes with a very basic but high quality network profiler showing basic statistics about the network such as the number of nodes, the number of edges, its density, its clustering coefficient, and other metrics. Automatically calculated node attributes such as node connectivity, clustering coefficient, betweenness centrality, or edge weight and similarly are trivial tasks and do not require long time to be calculated.

2.1.4. Editing. Gephi is highly interactive and provides clever shortcuts to highlight communities, and shortest paths or relative distances of any node to a node of interest are offered. Moreover, users can easily adjust or interactively filter the shapes and colors of the network's edges and nodes according to their attributes in order to reveal hidden patterns. It is not the purpose of this review to tutor how to use such applications as this can be found in the tool's relevant help pages. While Gephi is a great option for large-scale network visualization, manual network importing, multiple network handling, and manual node/edge/label editing can be tricky as many options are well hidden in Gephi's user interface or supported by specific plugins.

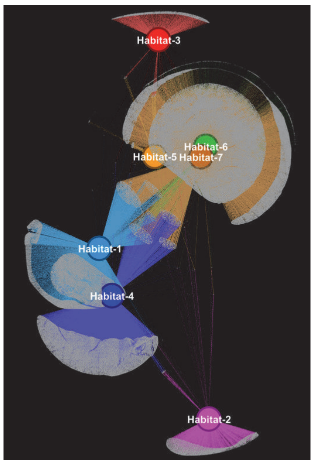

Figure 1: Gephi visualization of a network consisting of 202,424 nodes and 354,468 edges showing the distribution of 202,417 protein families across 7 habitats. A combination of OpenOrd and Yifan-Hu force-directed layout algorithm was used to calculate the node coordinates. Each habitat and its adjacent edges are colored uniquely. A step-by-step guide regarding the methods and the parametrization that were used is extensively described in the supplementary file.

2.1.5. File Formats. Gephi can load networks in GEXF, GDF, GML, GraphML, Pajek (NET), GraphViz (DOT), CSV, UCINET (DL), Tulip (TPL), Netdraw (VNA), and Excel spreadsheets. Similarly, Gephi can export networks in JSON, CSV, Pajek (NET), GUESS (GDF), Gephi (GEFX), GML, and GraphML [30] files. The easiest way to talk with Cytoscape is through GraphML formats, with Tulip through GEFX files and with Pajek through NET files. Unfortunately, in its current version, communication with other tools through other common file formats such as JSON fails.

2.1.6. Availability. Regardless of its very limited documentation, Gephi is a great, generic, nondedicated to biology, 2D network visualization tool. It mainly emphasizes fast and smooth rendering, fast layouting, efficient filtering, and interactive data exploration and we believe that it remains one of the best options for generic large-scale network visualization. A network example visualized by Gephi is shown in Figure 1. Gephi is available at: https://gephi.org/.

2.2. Tulip (Version 4.10.0). Tulip is one of the easiest-to-use network visualization tools and a decent option for visualization of larger-scale networks. Due to its simplicity, it is highly recommended for nonexperts as it comes with an easy-to-use interface. It is written in C++ and enables the development of algorithms, visual encodings, interaction techniques, data models, and domain-specific visualizations. Compared to other tools, it offers very appealing visualizations especially after enabling its great edge-bundling algorithm.

2.2.1. Scalability. In its current version, it is able to visualize thousands of nodes with hundreds of thousands of edges in an average computer and aims at becoming a great mediator between graph analysis and visualization. While Tulip is a top preference for medium-scale networks, to our experience it is not as scalable as Gephi.

2.2.2. Layouts. Its great plethora of layout algorithms makes it one of the best options for graph layout. At the moment, it supports simple (circular, random), force-directed (i.e., Fruchterman and Reingold [28], Kamada and Kawai [31]), hierarchical, multilevel, planar, and tree layout algorithms, most of them optimized and implemented within the Open Graph Drawing Framework (OGDF) [32]. As opposed to the more conservative force-directed layout algorithms, Fast Multipole Multilevel Layout is highly recommended for large-scale networks. While its layouts are of a great quality, in order to save time, the strategy to first calculate the nodes layout with Gephi or Pajek and then import to Tulip is highly recommended.

2.2.3. Postvisualization Analysis. By trying to bridge the gap between the analysis and visualization, Tulip comes with a rich pool of clustering and network topology analysis algorithms. Among others, Tulip currently implements the memory greedy but widely accepted greedy Markov Clustering (MCL) [29] as well as the fast and memory efficient Louvain Clustering [33] for unweighted graphs. In addition, Tulip incorporates various traditional algorithms for network exploration like algorithms to find the biconnected or the strongly connected components or algorithms dedicated to finding spanning trees or loops. Like before, for large-scale network analysis, running clustering algorithms externally is recommended.

In addition, Tulip comes with a very simple interface to ask topological questions. K-core decomposition of a graph, eccentricity centrality, degree, page rank, and betweenness centrality are few of the offered options and nodes' size or color can be adjusted according to a selected topological feature.

2.2.4. Editing. While Tulip does not come with a great variety of predefined color schemes, users can manually change the color, the size, and the shape of any node, label, or edge and save and reload the status of a network. Unfortunately, it can process one network per session and users must be careful as sometimes the visualization and the editing panels do not coordinate. Unfortunately, simple tasks such as interactively selecting the in/out edges of a node directly from the visualization can take significant amount of time.

2.2.5. Edge Bundling. While Tulip's renderer does not reach Gephi's or Cytoscape's resolution, it comes with one of the most appealing edge-bundling algorithms. Unfortunately, for large-scale network analysis, its edge-bundling algorithm can often become memory and CPU greedy so users must be patient. Finally saving the status of a bundled view compared to an unbundled view can lead to significantly higher storage requirements (see supplementary file for examples).

2.2.6. File Formats. It accepts as input simple tab delimited, Pajek, GEFX, GML, GraphViz, JSON, TLPB, and UCINET

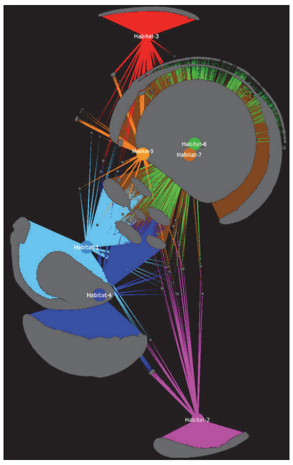

Figure 2: Tulip visualization of the same network like in Figure 1. The 7 habitats are highlighted and resized accordingly. An example of the same network after applying edge bundling is presented in the supplementary file. Nodes' coordinates were calculated using the Yifan-Hu layout algorithm from Gephi application.

files and exports to TLP, SVG, JSON, and GML formats. The easiest way to talk with Pajek is through NET files, with Cytoscape through GML or GraphML files, and with Gephi through GEFX files. Finally, Tulip comes with a very powerful generator of graphs of a user-defined size and topology.

2.2.7. Availability. Overall, Tulip is a generic 2D network visualization tool with a self-explanatory user interface and is suitable for large-scale node and edge layouting and analysis. A network example visualized by Tulip is shown in Figure 2. Tulip is available at: http://tulip.labri.fr/TulipDrupal/.

2.3. Cytoscape (Version 3.5.1). Cytoscape open-source Java application is the most widely used 2D network visualization tool in biology and health sciences. It supports all kinds of networks (e.g., weighted unweighted, bipartite, directed, undirected, and multiedged) and comes with an enormous library of additional plugins (>250). It was initially implemented to analyze molecular interaction networks and biological pathways and was aiming at integrating these networks with annotations, gene expression profiles, and other state data. Although Cytoscape was originally designed for biorelated research, now it serves as a generic platform for complex network analysis and visualization by providing a basic set of features for data integration, analysis, and visualization.

2.3.1. Scalability. Cytoscape implementations after version 3.0.0 come with tremendous rendering improvements, thus allowing Cytoscape to visualize large networks of hundred thousand nodes and edges. Despite these improvements, Cytoscape does not rank first for large-scale network analysis as it cannot scale significantly when it comes to analysis. Often Cytoscape's clustering and layout routines need great amount of memory and time. Therefore, for large-scale network analysis, it is suggested to run such processes in command line outside Cytoscape platform and load the results as node/edge attributes (groups in the case of clustering or coordinates in the case of a layout). In addition, Cytoscape is subject to Java's memory and running time limitations as most of its routines are implemented in Java.

2.3.2. Layouts. Like other tools, it comes with a very rich variety of simple (grid, random, and circular) or more sophisticated (force-directed, hierarchical) layout algorithms. Notably, for large-scale network analysis, users must be careful and change the default layout algorithm before creating a view. A simple grid or a simple circular layout is recommended as Cytoscape's force-directed layouts are memory and CPU greedy and the application might "hang." Another alternative could be OpenCL, one of the fastest layouts algorithms in Cytoscape. After version 3.2.0 OpenCL-based version is incorporated as a basic application. This layout is up to 100 times faster than the standard Prefuse layout and depends on the CyCL core app for OpenCL support. Nevertheless, calculating a first layout with Gephi or Pajek and then importing its results in Cytoscape can save time.

2.3.3. Postvisualization Analysis. Cytoscape is the most successful tool for bridging the gap between analysis and visualization and it comes with a great plethora of layout, clustering, and topological network analysis algorithms. ClusterMaker plugin [34], for example, includes attribute cluster algorithms such as AutoSOME Clustering [35] and Eisen's hierarchical and k-Means clustering [36] as well as topology-based clustering algorithms such as affinity propagation [37], community clustering (GLay) [38], MCODE [39], MCL, SCPS (Spectral Clustering of Protein Sequences) [40], and transitivity clustering [41]. Most clustering results can be visualized as a newly constructed network preserving the original edges, or as a heatmap. Like before, for large-scale network analysis, users are encouraged to run such algorithms externally.

In addition, Cytoscape incorporates one of the most advanced network profilers to explore network topological features. Users are able to view simple statistics like the average connectivity, betweenness centrality, clustering coefficient, and others. While such calculations are trivial for large-scale networks, plotting a topological feature against any other could be slow.

Finally, Cytoscape's latest versions incorporate a rather useful but slow and memory inefficient edge-bundling algorithm, not recommended for large-scale analysis.

2.3.4. Editing. Cytoscape is a protagonist in offering predefined visual styles and color schemes to create high quality and aesthetically beautiful visualizations. Its zooming and panning capabilities are very advanced and Cytoscape's satellite viewer makes it very easy for users to navigate and orient when the network is drawn outside the main canvas, something that is not trivial with Gephi. Finally, choosing adjacent nodes and edges from the UI is very responsive.

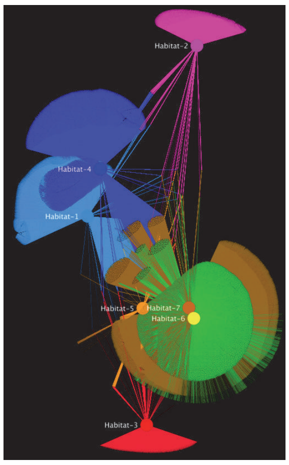

Figure 3: Cytoscape visualization of the same network like in Figure 1. The network consists of 202,424 nodes and 354,468 edges. The 7 habitats are colored accordingly. Like in Figure 2, coordinates were calculated using the Yifan-Hu layout algorithm from Gephi application.

2.3.5. File Formats. Cytoscape accepts many different input file formats such as its own CYS format, tab delimited, simple interaction file format (SIF), nested network format (NNF), graph markup language (GML), extensible graph markup and modelling language (XGMML), SBML [42], BioPAX [43], PSI-MI [44], GraphML, excel workbooks (.xls, .xlsx), and JSON. The easiest way to talk with Tulip and Gephi is through a GML format.

2.3.6. Availability. Overall, Cytoscape is the best visualization tool today for biological network analyses. Despite its user friendliness, its rich documentation, and the tremendous improvement of its user interface after version 3.0, familiarity with the tool and its available plugins still requires a steep learning curve for more advanced tasks. Cytoscape store currently hosts more than 250 plugins, specifically designed to address and automate complicated biological analyses. Plugins for functional enrichment, Gene Ontology annotations [45], gene name mapping, integration with biological public repositories, efficient online data retrieval, pathway analysis, direct network comparisons, differential expression, and statistical analysis make Cytoscape unique of its kind and therefore today it currently is and expected to remain the number-one player for biological network analysis. A network visualized by Cytoscape is shown in Figure 3. Cytoscape is available at http://www.cytoscape.org/.

Finally, CytoscapeWeb [46] and Cytoscape.js are separate projects. They are two very strong efforts aiming to incorporate Cytoscape's main visual functionalities in browser-based applications, something that of course is not suitable for large-scale network analysis. Users can use Cytoscape and export the networks in JSON format for Cytoscape.js.

2.4. Pajek (Version 5.01). Pajek is a generic, more than 20 years old, Microsoft Windows based network visualization tool, initially implemented for social network analysis, yet a very powerful application for analysis and visualization of massive networks.

2.4.1. Scalability. Pajek can easily visualize million nodes with billion connections in an average computer by outperforming any other available tool in the field. Pajek-XXL is a special implementation of Pajek with emphasis on huge scale network analysis. It needs at least 2-3 times less physical memory than Pajek and most of Pajek's memory intensive operations are optimized to be much faster. The main philosophy of Pajek-XXL is to extract smaller but most interesting and informative parts of a larger network which can be further analyzed and visualized with more advanced tools. The highest possible number of vertices that Pajek64-XXL can handle has been increased to 2 billion as for ordinary Pajek the limit is 100 million. Pajek-XXL uses 32-bit (4 bytes) integers for vertices numbers. Thus, the highest number of vertices that Pajek-XXL can handle is set to two billion. If network contains more vertices Pajek-3XL must be used. Pajek-3XL uses 64-bit (8 bytes) integers for vertices numbers. The highest number of vertices that Pajek-3XL can handle is currently set to 10 billion but can easily be further increased. Notably, the space needed to store a network in Pajek-3XL and Pajek-XXL is exactly the same.

2.4.2. Layouts. Graph layout, node merging, neighborhood detection, identification of strongly connected components, clique finding, manipulation of bipartite graphs, searching for shortest paths or maximum flows, clustering (i.e., Louvain), and computing centralities of vertices and centralizations of networks such as degree, closeness, betweenness, hubs and authorities, clustering coefficients, and Laplacian centrality are few of Pajek's capabilities. Notably, Pajek is memory efficient and very suitable for fast sparse network multiplication.

2.4.3. File Format. Pajek accepts very strict file input formats. The easiest way to talk with Tulip and Gephi is through a .net file

Pajek's user interface is simple, easy to get familiar with, and very responsive when it comes to analysis of massive networks. It was never intended to be the most advanced visualizer but it offers tremendous graph analyses methodologies thus making it a great candidate for analysis of massive networks and a great complement to the existing tools. A network example visualized by Pajek is shown in Figure 4. Pajek can be found at http://mrvar.fdv.uni-lj.si/pajek/.

3. Discussion

Despite the great plethora of available network visualization tools, due to the continuous increase of the data volume in health sciences, visualization and manipulation of large-scale networks with million nodes and edges still remain a bottleneck. While noninteractive libraries such as the Stanford Network Analysis Project (SNAP) [47], the outdated Large Graph Layout (LGL) [48], NetworkX [49], or the GraphViz [50] are preferred for backend calculations and large-scale static visualizations and while alternative network visualizations such as the ones offered by the Circos [51],

Table 1: Empirical evaluation of our top four interactive network visualization tools (Cytoscape, Gephi, Tulip, and Pajek) for large-scale biological network analysis.

	Cytoscape	Tulip	Gephi	Pajek
Scalability	**	*	* * *	* * **
User friendliness	**	* * **	* * *	*
Visual styles	* * **	**	* * *	*
Edge bundling	* * *	* * **	**	—
Relevance to biology	* * **	**	* * *	*
Memory efficiency	*	**	* * *	* * **
Clustering	* * **	* * *	*	**
Manual node/edge editing	* * *	* * **	* * *	*
Layouts	* * *	**	* * **	*
Network profiling	* * **	**	* * *	*
File formats	**	* * *	* * **	*
Plugins	* * **	**	* * *	*
Stability	* * *	*	* * **	* * *
Speed	**	*	* * *	* * **
Documentation	* * **	*	**	* * *

* = weaker; ** = medium; * * * = good; * * ** = strongest.

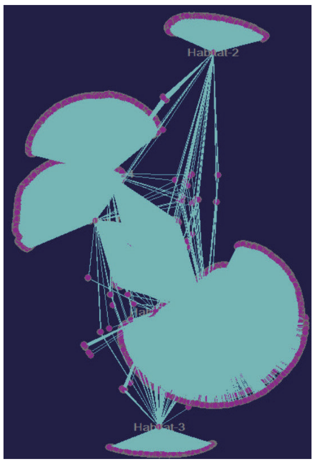

Figure 4: Pajek basic visualization of the same network like in Figure 1. The network consists of 202,424 nodes and 354,468 edges. Like in Figures 2 and 3, coordinates were calculated using the Yifan-Hu layout algorithm from Gephi application. Notably for a massive network it is highly recommended to first use Pajek layouting.

HivePlots [52], or BioFabric [53] can partially solve the hairball effect, the implementation of user friendly interactive tools to handle and visualize such large graphs still remains a very complicated task. Therefore, for the purposes of this review article, we tested several available standalone applications and concluded that Pajek, Tulip Gephi, and Cytoscape are top candidates for large-scale network visualization and analysis.

In conclusion, while Cytoscape is the best and the most preferred tool for biological analyses, it has scalability and memory issues and therefore it is not our top pick for large-scale network visualization. On the contrary, we rank it first for biological analyses as it is accompanied by a great plethora of more than 200 plugins. Compared to Tulip, Gephi, and Pajek, it has the richest palette of predefined color styles, the most efficient collection of clustering algorithms, and the best network profiler for intranetwork comparison of topological features.

Gephi clearly outperforms Cytoscape in terms of scalability and memory efficiency and, in our opinion, it is the best generic visualization tool for layouting large-scale networks. While it is fairly straightforward to use, sometimes node/edge editing options are well hidden in its user interface thus making it a bit confusing for the user. On the other hand, Gephi offers very advanced visualizations by allowing users to perform multiple tasks simultaneously, something that is not always easy with Cytoscape or Tulip. Overall, we rank Gephi as first when it comes to balance between large-scale network visualization and basic analysis.

Tulip is our third best option for large-scale network visualization. Its best characteristics are (i) the edge-bundling layout and (ii) its simplicity in editing the node's/edge's colors, labels, and attributes. Tulip is highly recommended for beginners due to its self-explanatory user interface.

Finally, Pajek and Pajek-XXL are the most scalable tools and highly recommended for basic visualizations of massive networks with >10 billion nodes, network sizes that Cytoscape, Tulip, and Gephi cannot handle in their current versions. Unfortunately, the lack of operating system interoperability as well as the lack of input file format flexibility and the lack of appealing visualizations prevent Pajek from being the top tool for advanced visualizations.

All the aforementioned observations are summarized in Table 1. Even though they may vary from user to user depending on the expertise and the case study, in our opinion, Cytoscape, Tulip, Pajek, and Gephi still remain the best large-scale network visualization and analysis tools in systems and network biology.

4. Conclusion

It is unfair and not straightforward to directly compare visualization tools with each other as they are implemented to serve different purposes. Nevertheless, as biological network sizes increase over time, combining the complementary advantages from different tools is a good strategy. While several file formats to describe the structure of network have been standardized, our experience showed that many of them cannot be properly exported or imported across several tools. In addition, even in the best cases where such an import/export problem is absent, often node and edge attributes cannot be transferred. Therefore, we believe that a catholic network converted to accurately convert a file format into any other by simultaneously keeping the maximum information about the network's components is mandatory. This way, switching between tools and various visualizations will become easier and more straightforward.

Additional Points

Availability of Data and Materials. The datasets used and/or analyzed during the current study are available from the corresponding author on reasonable request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Authors' Contributions

Georgios A. Pavlopoulos wrote the article, David Paez-Espino and Nikos C. Kyrpides provided the data for the figures, and Ioannis Iliopoulos supervised the study. All authors read and approved the final manuscript.

Acknowledgments

This work was supported by the US Department of Energy, Joint Genome Institute, a DOE Office of Science User Facility, under Contract no. DE-AC02-05CH11231, and used resources of the National Energy Research Scientific Computing Center, supported by the Office of Science of the US Department of Energy.

References

[1] G. A. Pavlopoulos, M. Secrier, C. N. Moschopoulos et al., "Using graph theory to analyze biological networks," *BioData Mining*, vol. 4, no. 1, article 10, 2011.

[2] G. A. Pavlopoulos, D. Malliarakis, N. Papanikolaou, T. Theodosiou, A. J. Enright, and I. Iliopoulos, "Visualizing genome and systems biology: Technologies, tools, implementation techniques and trends, past, present and future," *GigaScience*, vol. 4, no. 1, article no. 38, 2015.

[3] N. Gehlenborg, S. I. O'Donoghue, N. S. Baliga et al., "Visualization of omics data for systems biology," *Nature Methods*, vol. 7, no. 3, pp. S56–S68, 2010.

[4] S. I. O'Donoghue, A.-C. Gavin, N. Gehlenborg et al., "Visualizing biological data—now and in the future," *Nature Methods*, vol. 7, no. 3, pp. S2–S4, 2010.

[5] G. A. Pavlopoulos, E. Iacucci, I. Iliopoulos, and P. Bagos, "Interpreting the Omics 'era' Data," *Smart Innovation, Systems and Technologies*, vol. 25, pp. 79–100, 2013.

[6] G. A. Pavlopoulos, A. L. Wegener, and R. Schneider, "A survey of visualization tools for biological network analysis," *BioData Mining*, vol. 1, 12 pages, 2008.

[7] N. Bikakis, J. Liagouris, M. Krommyda, G. Papastefanatos, and T. Sellis, "GraphVizdb: A scalable platform for interactive large graph visualization," in *Proceedings of the 32nd IEEE International Conference on Data Engineering, ICDE 2016*, pp. 1342–1345, Helsinki, Finland, May 2016.

[8] J. Köhler, J. Baumbach, J. Taubert et al., "Graph-based analysis and visualization of experimental results with ONDEX," *Bioinformatics*, vol. 22, no. 11, pp. 1383–1390, 2006.

[9] F. Iragne, M. Nikolski, B. Mathieu, D. Auber, and D. Sherman, "ProViz: Protein interaction visualization and exploration," *Bioinformatics*, vol. 21, no. 2, pp. 272–274, 2005.

[10] Z. Hu, J.-H. Hung, Y. Wang et al., "VisANT 3.5: Multi-scale network visualization, analysis and inference based on the gene ontology," *Nucleic Acids Research*, vol. 37, no. 2, pp. W115–W121, 2009.

[11] E. Adar, "GUESS: a language and interface for graph exploration," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 791–800, Montreal, CA, USA, 2006.

[12] S. P. Borgatti, M. G. Everett, and L. C. Freeman, *Ucinet for Windows: Software for Social Network Analysis*, Analytic Technologies, Harvard, Mass, USA, 2002.

[13] O. Thimm, O. Bläsing, Y. Gibon et al., "MAPMAN: a user-driven tool to display genomics data sets onto diagrams of metabolic pathways and other biological processes," *Plant Journal*, vol. 37, no. 6, pp. 914–939, 2004.

[14] E. Demir, O. Babur, U. Dogrusoz et al., "PATIKA: An integrated visual environment for collaborative construction and analysis of cellular pathways," *Bioinformatics*, vol. 18, no. 7, pp. 996–1003, 2002.

[15] G. A. Pavlopoulos, S. D. Hooper, A. Sifrim, R. Schneider, and J. Aerts, "Medusa: A tool for exploring and clustering biological networks," *BMC Research Notes*, vol. 4, article no. 384, 2011.

[16] B. J. Breitkreutz, C. Stark, and M. Tyers, "Osprey: a network visualization system," *Genome Biology*, vol. 4, article R22, no. 3, 2003.

[17] M. Secrier, G. A. Pavlopoulos, J. Aerts, and R. Schneider, "Arena3D: visualizing time-driven phenotypic differences in biological systems," *BMC Bioinformatics*, vol. 13, no. 1, article 45, 2012.

[18] G. A. Pavlopoulos, S. I. O'Donoghue, V. P. Satagopam, T. G. Soldatos, E. Pafilis, and R. Schneider, "Arena3D: visualization of biological networks in 3D," *BMC Systems Biology*, vol. 2, article 104, 2008.

[19] A. Theocharidis, S. van Dongen, A. J. Enright, and T. C. Freeman, "Network visualization and analysis of gene expression data using BioLayout Express (3D)," *Nature Protocols*, vol. 4, no. 10, pp. 1535–1550, 2009.

[20] P. Shannon, A. Markiel, O. Ozier et al., "Cytoscape: a software Environment for integrated models of biomolecular interaction networks," *Genome Research*, vol. 13, no. 11, pp. 2498–2504, 2003.

[21] D. Auber, "Tulip —a huge graph visualization framework," in *Graph Drawing Software*, M. Jünger and P. Mutzel, Eds., Mathematics and Visualization, pp. 105–126, Springer, Berlin, Germany, 2004.

[22] M. Jacomy, T. Venturini, S. Heymann, and M. Bastian, "ForceAtlas2, a continuous graph layout algorithm for handy network visualization designed for the Gephi software," *PLoS ONE*, vol. 9, no. 6, Article ID e98679, 2014.

[23] A. Mrvar and V. Batagelj, "Analysis and visualization of large networks with program package Pajek," *Complex Adaptive Systems Modeling*, vol. 4, no. 6, 2016.

[24] V. Batagelj and A. Mrvar, "Pajeka— program for large network analysis," *Connections*, vol. 21, no. 2, pp. 47–57, 1998.

[25] I. A. Chen, V. M. Markowitz, K. Chu et al. et al., "IMG/M: integrated genome and metagenome comparative data analysis system," *Nucleic Acids Research*, 2016.

[26] S. Martin, W. M. Brown, R. Klavans, and K. W. Boyack, "OpenOrd: An open-source toolbox for large graph layout," in *Proceedings of the Visualization and Data Analysis 2011*, San Francisco Airport, Calif, USA, January 2011.

[27] H. Yifan, "Efficient, high-quality force-directed graph drawing," *The Mathematica Journal*, vol. 10, no. 1, 2006.

[28] T. M. J. Fruchterman and E. M. Reingold, "Graph drawing by force-directed placement," *Software—Practice and Experience*, vol. 21, no. 11, pp. 1129–1164, 1991.

[29] A. J. Enright, S. Van Dongen, and C. A. Ouzounis, "An efficient algorithm for large-scale detection of protein families," *Nucleic Acids Research*, vol. 30, no. 7, pp. 1575–1584, 2002.

[30] U. Brandes, M. Eiglsperger, J. Lerner, and C. Pich, "Graph markup language (GraphML)," in *Handbook of Graph Drawing and Visualization*, pp. 517–541, 1999.

[31] T. Kamada and S. Kawai, "An algorithm for drawing general undirected graphs," *Information Processing Letters*, vol. 31, no. 1, pp. 7–15, 1989.

[32] M. Chimani, C. Gutwenger, M. Jünger, G. W. Klau, and K. Klein, *The Open Graph Drawing Framework (OGDF)*, Chapman & Hall, London, UK, 2014.

[33] V. D. Blondel, J. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, Article ID P10008, 2008.

[34] J. H. Morris, L. Apeltsin, A. M. Newman et al., "ClusterMaker: a multi-algorithm clustering plugin for Cytoscape," *BMC Bioinformatics*, vol. 12, article 436, 2011.

[35] A. M. Newman and J. B. Cooper, "AutoSOME: A clustering method for identifying gene expression modules without prior knowledge of cluster number," *BMC Bioinformatics*, vol. 11, article no. 117, 2010.

[36] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein, "Cluster analysis and display of genome-wide expression patterns," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 95, no. 25, pp. 14863–14868, 1998.

[37] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *American Association for the Advancement of Science. Science*, vol. 315, no. 5814, pp. 972–976, 2007.

[38] M. E. J. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, vol. 69, no. 2, Article ID 026113, pp. 1–26113, 2004.

[39] G. D. Bader and C. W. Hogue, "An automated method for finding molecular complexes in large protein interaction networks," *BMC Bioinformatics*, vol. 4, no. 2, 2003.

[40] T. Nepusz, R. Sasidharan, and A. Paccanaro, "SCPS: A fast implementation of a spectral method for detecting protein families on a genome-wide scale," *BMC Bioinformatics*, vol. 11, article no. 120, 2010.

[41] T. Wittkop, D. Emig, S. Lange et al., "Partitioning biological data with transitivity clustering," *Nature Methods*, vol. 7, no. 6, pp. 419-420, 2010.

[42] M. Hucka, A. Finney, H. M. Sauro et al., "The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models," *Bioinformatics*, vol. 19, no. 4, pp. 524–531, 2003.

[43] J. S. Luciano and R. D. Stevens, "E-Science and biological pathway semantics," *BMC Bioinformatics*, vol. 8, no. 3, article no. S3, 2007.

[44] H. Hermjakob, L. Montecchi-Palazzi, G. Bader et al., "The HUPO PSI's Molecular Interaction format—a community standard for the representation of protein interaction data," *Nature Biotechnology*, vol. 22, no. 2, pp. 177–183, 2004.

[45] M. Ashburner, C. A. Ball, J. A. Blake et al., "Gene ontology: tool for the unification of biology," *Nature Genetics*, vol. 25, no. 1, pp. 25–29, 2000.

[46] C. T. Lopes, M. Franz, F. Kazi, S. L. Donaldson, Q. Morris, and G. D. Bader, "Cytoscape web: An interactive web-based network browser," *Bioinformatics*, vol. 26, no. 18, Article ID btq430, pp. 2347-2348, 2010.

[47] J. Leskovec and R. Sosi, "SNAP: a general-purpose network analysis and graph-mining library," *ACM Transactions on Intelligent Systems and Technology*, vol. 8, no. 1, pp. 1–20, 2016.

[48] A. T. Adai, S. V. Date, S. Wieland, and E. M. Marcotte, "LGL: Creating a map of protein function with an algorithm for visualizing very large biological networks," *Journal of Molecular Biology*, vol. 340, no. 1, pp. 179–190, 2004.

[49] A. Hagberg, D. Schult, and P. Swart, "Exploring Network Structure, Dynamics, and Function using Network," in *Proceedings of the 7th Python in Science Conference (SciPy 2008)*, pp. 11–15, 2008.

[50] E. R. Gansner and S. C. North, "An open graph visualization system and its applications to software engineering," *Software—Practice & Experience*, vol. 30, no. 11, pp. 1203–1233, 2000.

[51] M. Krzywinski, J. Schein, I. Birol et al., "Circos: An information aesthetic for comparative genomics," *Genome Research*, vol. 19, no. 9, pp. 1639–1645, 2009.

[52] M. Krzywinski, I. Birol, S. J. Jones, and M. A. Marra, "Hive plots-rational approach to visualizing networks," *Briefings in Bioinformatics*, vol. 13, no. 5, pp. 627–644, 2012.

[53] W. J. R. Longabaugh, "Combing the hairball with BioFabric: A new approach for visualization of large networks," *BMC Bioinformatics*, vol. 13, no. 1, article no. 275, 2012.