NAViGatOr: Network Analysis, Visualization and Graphing Toronto

Kevin R. Brown¹, David Otasek¹, Muhammad Ali¹, Michael J. McGuffin², Wing Xie¹, Balju Devani¹, Ian Lawson van Toch¹ and Igor Jurisica¹,²,*

¹Division of Signaling Biology, Ontario Cancer Institute and the University Health Network, Toronto, ON, ²Department of Software and IT Engineering, École de technologie supérieure, Montreal, QC and
²Departments of Medical Biophysics and Computer Science, University of Toronto, Toronto, ON, USA

Received on July 22, 2009; revised on September 21, 2009; accepted on October 11, 2009
Advance Access publication October 16, 2009
Associate Editor: Jonathan Wren

© The Author(s) 2009. Published by Oxford University Press.

While many of these tools are effective and widely used, there are several critical areas where these applications require improvement (reviewed in Suderman and Hallett, 2007). Scalability is essential to visualize the tens of thousands of known PPI, which is a challenge for current layout algorithms and software. Biological graph drawing software must also be able to handle richly annotated data, including genomic and proteomic profiles, KEGG pathways (Kanehisa and Goto, 2000), Gene Ontology (GO) annotations, data in PSI-MI (Hermjakob et al., 2004b) and BioPAX formats (http://www.biopax.org/), in addition to the vast quantity of microarray data that is currently available.

NAViGatOr builds upon these earlier efforts, addressing known issues in existing software. NAViGatOr uses a combination of hardware-based graphics acceleration and highly optimized layout algorithms to enable interactive visualization of large networks. It supports community-based data interchange formats, such as PSI-MI, BioPAX and GML, facilitating interoperability with existing software tools. Additionally, NAViGatOr includes a rich suite of built-in analysis and visualization functions, which can be extended through an application programming interface (API). Here, we describe the implementation of NAViGatOr, and highlight how this tool improves upon existing network visualization packages.

2 SOFTWARE

2.1 Implementation

NAViGatOr has been implemented in Java (v1.6), providing platform-independence, and uses JOGL (https://jogl.dev.java.net/) to enable OpenGL hardware-accelerated graphics rendering. At present, the core code-base is closed-source to ensure stability, but future enhancements will extend the plug-in API to an OSGi-compliant framework that enables community-driven extensibility.

2.2 Features

NAViGatOr enables interactive visualization and analysis of complex graphs that are typical in systems biology studies. Graphs can be loaded from PSI-MI XML, BioPAX, GML and tab-delimited text files, or through online databases such as I2D (http://ophid.utoronto.ca/i2d) and cPATH (http://cbio.mskcc.org/cpath/).

*To whom correspondence should be addressed.

To whom correspondence should be addressed.

The availability of protein–protein interaction (PPI) data is increasing rapidly through literature-derived databases (Bader et al., 2003; Breitkreutz et al., 2002; Hermjakob et al., 2004a; Peri et al., 2004; Xenarios et al., 2000; Zanzeni et al., 2002), high-throughput detection methods (Barrios-Rodiles et al., 2005; Rual et al., 2005) and computational predictions (Brown and Jurisica, 2005; Persico et al., 2005). These data, collectively referred to as the interactome, are critical to our understanding of both normal cellular processes and disease mechanisms. Visualizing the interactome, along with integrating orthogonal data types, may aid in the understanding of cell function, help elucidate hidden relationships within the data and help prioritize functional studies.

Several biological graph visualization tools are currently available, implementing a diverse range of approaches and algorithms (Breitkreutz et al., 2003; Chin et al., 2008; Hu et al., 2004; Ju and Han, 2003; Macpherson et al., 2009; Paaneman and Wong, 2009). Cytoscape (Shannon et al., 2003), in particular, has been widely adopted by the biological community for its ease of use and extensibility through open source plug-in development.

1 INTRODUCTION

The availability of protein–protein interaction (PPI) data is increasing rapidly through literature-derived databases (Bader et al., 2003; Breitkreutz et al., 2002; Hermjakob et al., 2004a; Peri et al., 2004; Xenarios et al., 2000; Zanzeni et al., 2002), high-throughput detection methods (Barrios-Rodiles et al., 2005; Rual et al., 2005) and computational predictions (Brown and Jurisica, 2005; Persico et al., 2005). These data, collectively referred to as the interactome, are critical to our understanding of both normal cellular processes and disease mechanisms. Visualizing the interactome, along with integrating orthogonal data types, may aid in the understanding of cell function, help elucidate hidden relationships within the data and help prioritize functional studies.

Several biological graph visualization tools are currently available, implementing a diverse range of approaches and algorithms (Breitkreutz et al., 2003; Chin et al., 2008; Hu et al., 2004; Ju and Han, 2003; Macpherson et al., 2009; Paaneman and Wong, 2009). Cytoscape (Shannon et al., 2003), in particular, has been widely adopted by the biological community for its ease of use and extensibility through open source plug-in development.

While many of these tools are effective and widely used, there are several critical areas where these applications require improvement (reviewed in Suderman and Hallett, 2007). Scalability is essential to visualize the tens of thousands of known PPI, which is a challenge for current layout algorithms and software. Biological graph drawing software must also be able to handle richly annotated data, including genomic and proteomic profiles, KEGG pathways (Kanehisa and Goto, 2000), Gene Ontology (GO) annotations, data in PSI-MI (Hermjakob et al., 2004b) and BioPAX formats (http://www.biopax.org/), in addition to the vast quantity of microarray data that is currently available.

NAViGatOr builds upon these earlier efforts, addressing known issues in existing software. NAViGatOr uses a combination of hardware-based graphics acceleration and highly optimized layout algorithms to enable interactive visualization of large networks. It supports community-based data interchange formats, such as PSI-MI, BioPAX and GML, facilitating interoperability with existing software tools. Additionally, NAViGatOr includes a rich suite of built-in analysis and visualization functions, which can be extended through an application programming interface (API). Here, we describe the implementation of NAViGatOr, and highlight how this tool improves upon existing network visualization packages.

2 SOFTWARE

2.1 Implementation

NAViGatOr has been implemented in Java (v1.6), providing platform-independence, and uses JOGL (https://jogl.dev.java.net/) to enable OpenGL hardware-accelerated graphics rendering. At present, the core code-base is closed-source to ensure stability, but future enhancements will extend the plug-in API to an OSGi-compliant framework that enables community-driven extensibility.

2.2 Features

NAViGatOr enables interactive visualization and analysis of complex graphs that are typical in systems biology studies. Graphs can be loaded from PSI-MI XML, BioPAX, GML and tab-delimited text files, or through online databases such as I2D (http://ophid.utoronto.ca/i2d) and cPATH (http://cbio.mskcc.org/cpath/). Both
Fig. 1. Screen capture of the NA ViGaTOR user interface. Labels indicate the various tools and descriptive regions of the interface. A graph is shown in the ‘Graph Panel’, with edges adjusted automatically by ‘Edge Filters’. Filters can be used to automatically control visual attributes of both nodes and edges.

2D and 3D network views are fully interactive, allowing the user to manually manipulate the graph, or to use automated layouts such as circular, linear or concentric circular on subsets of nodes or entire graphs. A spreadsheet view supports selecting and deselecting nodes, edges and paths based on any attributes. Nodes and edges can be grouped into subsets, which can be collapsed or expanded to simplify views, or manipulated through set operations. Network analysis tools provide information about node and edge connectivity, shortest paths, identify hubs, cliques and articulation points and summarize network statistics. NA ViGaTOR can also use a multi-threaded implementation to efficiently generate random networks for enrichment analyses. Fully annotated graphs can be exported to six different graphics formats, including PDF and SVG. In summary, NA ViGaTOR provides a network analysis platform that is rich in the features essential to many biological applications, and yet is extensible through a plug-in interface to include additional features when required. See Figure 1 and the Supplementary Materials for examples of the NA ViGaTOR interface and rendered networks.

2.3 Advances
NA ViGaTOR’s ability to handle larger datasets is facilitated through optimized layout algorithms, hardware-based graphics acceleration and a reduced memory footprint relative to other software. NA ViGaTOR performs an initial layout using Graph Drawing with Intelligent Placement (GRIP; Gajer and Kobourov, 2002), which performs network layout in near linear time, and then continuously updates the layout of the graph using a multi-threaded grid-variant (Fruchterman and Reingold, 1991) of a force-directed layout algorithm. When benchmarked against the force-directed algorithms in Cytoscape and VisANT, NA ViGaTOR consistently provided graphs rendered in significantly shorter time (Fig. 2; Supplementary Fig. 3.3). Only the yFiles Organic plug-in for Cytoscape rendered in similar time to NA ViGaTOR, although the resulting graph was poorly optimized (compare Supplementary Fig. 3.5C to Supplementary Fig. 3.4C).

OpenGL enables NA ViGaTOR to take advantage of hardware-based acceleration to render larger graphs in both 2D and 3D. Additionally, the data structures within NA ViGaTOR were designed to maintain a small memory footprint in order to provide greater scalability for large datasets. When compared against Cytoscape and VisANT, NA ViGaTOR had a memory footprint approximately half that of Cytoscape, although a 12–38% larger footprint than VisANT (Supplementary Fig. 5.1).

The NA ViGaTOR user interface includes unique tools to help simplify the ‘hairball’, which is a common challenge in many PPI...
for supporting the I2D database, which provides PPI data and
bars were used to show the cumulative loading and rendering time, or the
PSI-MI XML v1.0 file, and Osprey required a tab-delimited text file. Stacked
Interviewer3 required a GML export from NA ViGaTOR, VisANT required a
Only Cytoscape and NA ViGaTOR were able to load the BioPax file directly;
also serve as a platform to explore novel ways for biologists to
protecting the core code-base of the application. NA ViGaTOR will
driven development through small, tightly coupled bundles while
version of NA ViGaTOR includes a plug-in API, NA ViGaTOR 3.0
to a more versatile, comprehensive platform. While the current
NA ViGaTOR has evolved from an in-house visualization tool
complicated networks.
its radius in edges, allow users to easily select specific subsets
to (de)select a connected neighborhood of nodes by dragging out
areas of the network and focus on important areas by ‘fading out’
networks. Alpha blending is a technique to de-emphasize unimportant
3 FUTURE DEVELOPMENT
NAViGaTOR has evolved from an in-house visualization tool to a more versatile, comprehensive platform. While the current version of NAViGaTOR includes a plug-in API, NAViGaTOR 3.0 will adopt a more formal open plug-in interface compliant with the OSGi framework. This framework will allow for community-driven development through small, tightly coupled bundles while protecting the core code-base of the application. NAViGaTOR will also serve as a platform to explore novel ways for biologists to interact with graphs, as well as new ways to encode and display information in biological networks.

ACKNOWLEDGEMENTS
We would like to acknowledge Richard Lu and Frederic Breaud for supporting the I2D database, which provides PPI data and
Annotations for NAViGaTOR, Rick Valenzano for researching and implementing the GRIP layout, and Uzma Khan for helping with updating and improving the NAViGaTOR web site (http://ophid.utoronto.ca/navigator/).

Funding: Genome Canada via the Ontario Genomics Institute; Canada Foundation for Innovation (grant nos 12301 and 20383); Canada Research Chair Program in part; Ontario Research Fund Research Excellence.

Conflict of Interest: none declared

REFERENCES