Lessons learned from designing and implementing Network Explorer, a real world network visual analytics tool using open source software

John Alexis Guerra-Gomez*
Computing and Systems Departament, Universidad de los Andes †

ABSTRACT

This paper describes some of the lessons learned by designing, implementing, and deploying in real world scenarios a web-based network exploration tool called Network Explorer. Presented initially in [11], Network Explorer was designed based on the expressed needs of PARC's clients and later deployed it as part of a larger system for fraud detection in health care. Thanks to the extensive use of open source software, Network Explorer offers state of the art information visualization techniques for network analysis. Therefore, to contribute back to the community we released two open source libraries: an in-browser network clustering library, and a dynamic group-in-a-box layout algorithm that distributes nodes on the screen based on their clusters. These two libraries allow Network Explorer's users to recompute and navigate network clusters based on dynamic filtering.

Index Terms: H.1.2 [Information Systems]: User/Machine Systems—Human information processing H.5.2 [INFORMATION INTERFACES AND PRESENTATION (e.g., HCI)]: User Interfaces—

1 INTRODUCTION

The information visualization research community constantly produces great advancements that could greatly benefit the industry and government sectors. However, implementing real world deployments of such advancements is a very difficult endeavor. Open source software packages of research projects significantly ease the difficulty of these implementation tasks. They allow companies and governments to benefit from already developed and sometimes tested software. Moreover, these companies and organizations can return to the community some of the best practices they learned by producing open source packages on their own. In this paper we present some of the best practices we learned by developing Network Explorer, a web based visual analytics tool, created to support the specific tasks required by our customers at Xerox Parc. We also present our contribution to the community by the means of two open source packages that were developed to allow Network Explorer users to dynamically recompute and navigate communities of suspicious doctors or interrelated police officers.

Whereas many network visualization tools existed by the time of this project, such as Gephi [3], Cytoscape [18], Vizter [12], Lanet-vi [6], Pajek [5], NodeTrix [14], NodeXL [7], Tulip [2, 4] and GCV [20, 1], none of them fulfilled all of the requirements of our project. We needed a tool that was web-based, scalable, robust enough to support an industry setting, and more importantly, which could be integrated as part of our current analytics solution. We did however built on top of the wonderful d3.js [8] library and implement state of the art network visualization concepts when possible.

*e-mail: ja.guerrag@uniandes.edu.co
†This work was developed when the author was part of the Palo Alto Research Center

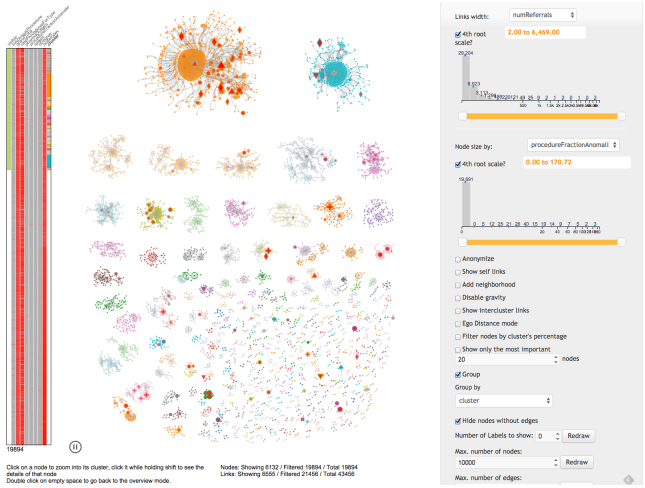

Figure 1: Network of 19,894 doctors and pharmacies that interact through prescriptions in a medical system. Network Explorer's rank-by-feature shows the top 6,132 most important nodes for the analyst, according to the level of suspiciousness of each node and its connectivity. Network Explorer offers the option for analysts to hide edges between clusters for legibility.

2 NETWORK EXPLORER

The first lesson we learned developing Network Explorer was that our users needed to analyze networks in two ways: **getting overviews** (Which correspond *overview* and *topology* tasks from the task taxonomy of graph visualization [15]), and **exploring the connections of a small set of interesting nodes** (*Attribute-based* and *browsing* tasks from the taxonomy).

To address these needs, Network Explorer implements two main modes: the *Overview mode* and the *ego-centric mode*.

2.1 Overview mode

The overview mode enables users to obtain a summary of the network structure. However, providing summaries of hundreds of thousands of nodes is more than challenging. Because of this we had to implement a rank-by-feature algorithm that selected the most important nodes to display on the overview. Unfortunately, selecting the top 1.000 nodes out of 100.000 would many times lead to 1.000 unconnected nodes that won't represent the structure of the network. To address this, we implemented an heuristic that selected a smaller core of the top nodes of the graph, and then added a subset of their most significant neighbors. With this we learned that **it is important to provide semantically meaningful summaries of the networks that take into account node's connections**.

Given that our users wanted to identify communities of suspicious doctors working together to hack the system, or clusters of police officers that interacted better together, we added community detection algorithms to infer clusters out of our networks. To illustrate this clusters we initially used color hues, but this wasn't enough to help our customers understand the characteristics of these

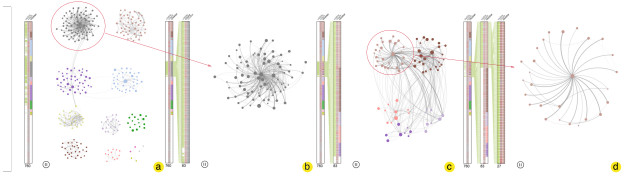

Figure 2: Users can jump into clusters to focus their exploration on specific parts of the network. In the figure, the user has decided to jump into the gray cluster on the top left corner of (a). Afterwards, only the 83 nodes in the cluster are shown on the screen, as shown in (b). Then the user reruns the clustering algorithm to detect sub-communities as shown in (c). Note how the cluster column in the Node Navigator in (c) reflects the new clusters; see Figure 6 for details of the cluster column. Finally the user decides to jump in again into one of the newly detected sub-clusters with 27 nodes (d). The Node Navigator on the left of (d) shows that the user has jumped into a cluster two times.

communities. To improve on this, we added a dynamic group-in-a-box algorithm that distributes communities on the screen, allowing users to understand the topology of the communities, while still identifying the actors that connected clusters together. This was so successful with our users that we released this algorithm as an easy to use d3.js [8] open source plugin [9], and we learned that **dynamic layouts that separate clusters on the screen helped our users understanding communities.** Our algorithm improves over the original group-in-a-box [17, 19] by distributing the communities on the screen using animations, and keeping some forces between the clusters, which highlight nodes that connect different communities.

Once our users were able to identify clusters, they wanted to explore further their internal connections. In our problem domains, these clusters could contain hundreds of nodes and were complicated networks on their own. To ease the understanding of those networks, we enabled users to *jump into* clusters by clicking on them, which filtered out all the other nodes and left only the selected cluster on the screen. Then, users could further filter down by node or edge attributes and recompute clusters on the remaining nodes thanks to an in-browser clustering library. The process is illustrated in Figure 2. Given the success of this feature with our users we decided to open source it too as a JavaScript library called NetClustering.js [10]. With this we learned that **allowing users to explore clusters and sub-clusters of dynamic selections helped them understanding complex networks**.

With large networks, navigating into clusters and sub-clusters proved to be disorienting. To address this, we developed a visualization widget called the Node Navigator that helped users keeping track of where in the graph they were. Shown in Figure 2, the Node Navigator displays the proportion of shown nodes vs filtered out, other node's attributes including their cluster, and how many times the user has jumped into a cluster. With the Node Navigator we learned that **is important keeping users informed of what parts of the networks they are exploring**.

2.2 The ego-centric mode

The second mode of usage of Network explorer is the ego-centric mode. This mode allows users to start exploring a network from a one or a few nodes of interests. Then, users can start **expanding those nodes' connections on demand** to find interesting relationships. This was especially useful for our customers because it allowed them to find suspicious doctors that heavily interacted with the same pharmacies used by known misbehaving practitioners. The ego-centric mode makes use of an algorithm inspired by the degree-of-interest trees [13, 16] that ease the exploration process while supporting details on demand.

Initially, the ego-centric mode uses an especial layout that fixes nodes by their distance to the nodes of interests, supporting the navigation task. However, users could switch back to the traditional force-directed layout and pin nodes to the screen, allowing the exploration of common paths between selected nodes. This taught us that **allowing different layouts help support different navigation tasks.**

The development and deployment of Network Explorer brought us many lessons learned that we try to convey on this paper, such that they can be of any use for future similar projects. This project wouldn't have been possible without the help of the open source projects we used, and we can give back to the community with our libraries.

REFERENCES

[1] CGV - Coordinated Graph Visualization, 2015.

[2] Data Visualization Software — Tulip, 2015.

[3] Gephi - The Open Graph Viz Platform. https://gephi.github.io/, 2015.

[4] D. Auber, D. Archambault, R. Bourqui, A. Lambert, M. Mathiaut, P. Mary, M. Delest, J. Dubois, and G. Melançon. The Tulip 3 Framework: A Scalable Software Library for Information Visualization Applications Based on Relational Data. Technical report, jan 2012.

[5] V. Batagelj and A. Mrvar. *Pajek - analysis and visualization of large networks*. Springer, 2004.

[6] M. G. Beiró, J. I. Alvarez-Hamelin, and J. R. Busch. A low complexity visualization tool that helps to perform complex systems analysis. *New Journal of Physics*, 10, 2008.

[7] E. M. Bonsignore, C. Dunne, D. Rotman, M. Smith, T. Capone, D. L. Hansen, and B. Shneiderman. First Steps to NetViz Nirvana: Evaluating Social Network Analysis with NodeXL. 2009.

[8] M. Bostock, V. Ogievetsky, and J. Heer. D3 Data-Driven Documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309, dec 2011.

[9] J. A. Guerra-Gomez. Force-in-a-box Dynamic group-in-a-box implementation https://github.com/john-guerra/forceInABox, 2014.

[10] J. A. Guerra-Gomez. NetClustering.js. https://github.com/john-guerra/netClusteringJs, 2014.

[11] J. A. Guerra-Gomez, A. Wilson, J. Liu, D. Davies, P. Jarvis, and E. Bier. Network Explorer. In *Proceedings of the International Working Conference on Advanced Visual Interfaces - AVI '16*, pages 108–111, New York, New York, USA, 2016. ACM Press.

[12] J. Heer and D. Boyd. Vizster: Visualizing online social networks. ... *Visualization, 2005. INFOVIS 2005. IEEE* ..., 2005.

[13] J. Heer and S. K. Card. DOITrees revisited: scalable, space-constrained visualization of hierarchical data. In *Proceedings of the working conference on Advanced visual interfaces - AVI '04*, page 421, New York, New York, USA, may 2004. ACM Press.

[14] N. Henry, J. Fekete, and M. McGuffin. NodeTrix: a hybrid visualization of social networks. *Visualization and Computer* ..., 2007.

[15] B. Lee, C. Plaisant, C. S. Parr, J.-D. Fekete, and N. Henry. Task taxonomy for graph visualization. In *Proceedings of the 2006 AVI workshop on BEyond time and errors novel evaluation methods for information visualization - BELIV '06*, page 1, New York, New York, USA, may 2006. ACM Press.

[16] D. Nation, D. Roberts, and S. Card. Browse Hierarchical Data with the Degree of Interest Tree. *submitted to CHI*, 2002.

[17] E. M. Rodrigues, N. Milic-Frayling, M. Smith, B. Shneiderman, and D. Hansen. Group-in-a-Box Layout for Multi-faceted Analysis of Communities. In *2011 IEEE Third Int'l Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third Int'l Conference on Social Computing*, pages 354–361. IEEE, oct 2011.

[18] P. Shannon, A. Markiel, and O. Ozier. Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome* ..., 2003.

[19] B. Shneiderman and C. Dunne. Interactive network exploration to derive insights: filtering, clustering, grouping, and simplification. *Graph Drawing*, 2013.

[20] C. Tominski, J. Abello, and H. Schumann. CGV - An interactive graph visualization system. *Computers & Graphics*, 2009.