Carina: Interactive Million-Node Graph Visualization using Web Browser Technologies

Dezhi Fang, Matthew Keezer, Jacob Williams, Kshitij Kulkarni,
Robert Pienta, Duen Horng Chau
Georgia Institute of Technology
{dezhifang, rkeezer3, jake.williams, kkulkarni36, pientars, polo}@gatech.edu

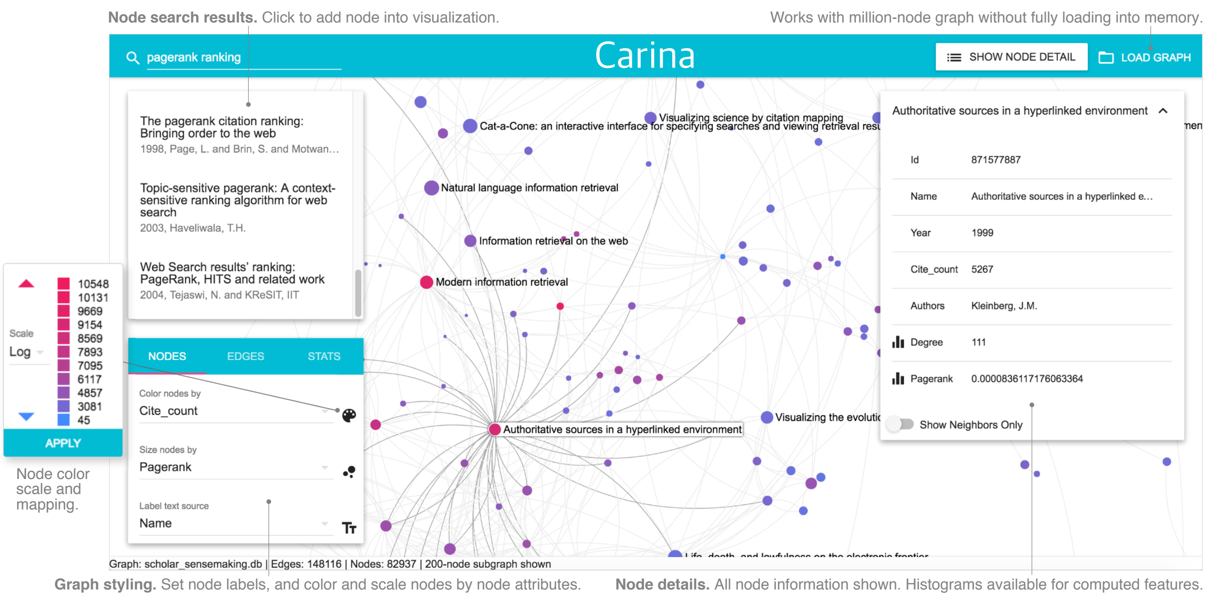

Figure 1: Carina visualizing citation network data (83k nodes, 150k edges) [2]. Carina works with out-of-core million-node graphs (up to 69M edges). Carina uses latest web browser technologies for high-performance graph rendering (WebGL), easy cross-platform deployment (Electron), and lightweight, scalable data storage.

ABSTRACT

We are working on a scalable, interactive visualization system, called Carina, for people to explore million-node graphs. By using latest web browser technologies, Carina offers fast graph rendering via WebGL and works across desktop (via Electron) and mobile platforms. Different from most existing graph visualization tools, Carina does not store the full graph in RAM, enabling it to work with graphs with up to 69M edges. We are working to improve and open-source Carina, to offer researchers and practitioners a new, scalable way to explore and visualize large graph datasets.

Keywords

Interactive graph visualization, web standards, WebGL

WWW 2017, April 3–7, 2017, Perth, Australia.
ACM 978-1-4503-4913-0/17/04.
http://dx.doi.org/10.1145/3041021.3054234

1. INTRODUCTION

Large graph data have become increasingly common. Visualizing graphs can help people more easily understand relations among entities. However, existing graph visualization systems face multiple technical, visual, and scalability challenges.

Visual Scalability Challenge. Most graph visualization systems follow the convention approach of visualizing the entire graph (e.g., Gephi [1], Cytoscape [5]). For million-node graphs, such visualizations generate "hairballs" with extreme edge crossings [2], overwhelming human perception and impeding understanding.

Data Scalability Challenge. In most existing systems, a graph dataset must first be fully loaded into memory. The available RAM becomes a barrier for analyzing larger graphs with million nodes or more. For example, the popular Gephi system runs out of RAM when trying to load the LiveJournal social graph with 69M edges [4].

Technology Challenge. Most existing visualization tools are built primarily for desktop use, precluding analysis in mobile environments, which are increasingly common (e.g., DARPA GUARD DOG mobile graph analytics [3]). For instance, Gephi and Cytoscape are built using Java and run only on desktop computers.

We present our ongoing research to tackle the above challenges.

- We introduce Carina (Fig.), a cross-platform prototype graph visualization system. Using latest web browser technologies, Carina offers fast graph rendering via WebGL, and lightweight, cross-platform deployment via Electron.
- We demonstrate that Carina works with million-node graphs (up to 69M edges), without requiring them to fit in RAM, unlike most existing graph visualization tools. Carina visualizes user-specified subgraphs (instead of the whole graph), allowing users to focus their exploration on the most relevant graph regions. Carina uses SQLite for storing and querying out-of-core datasets.

2. CARINA FEATURE HIGHLIGHTS

High-performance Graph Rendering via WebGL. Many visualization libraries, e.g., d3.js (https://d3js.org) and processing.js, render graphs using technologies like *Scalable Vector Graphics* (SVG) and HTML Canvas elements. However, their rendering speed (screen refresh rate) begins to deteriorate significantly starting at low hundreds of nodes and edges (https://github.com/anvaka/graph-drawing-libraries).

We have identified WebGL as a high-performance graphics technology that has strong potential for fast graph rendering. WebGL uses GPU acceleration, and is supported by all modern web browsers (http://caniuse.com/#feat=webgl). Carina leverages WebGL's high-speed graphics capabilities through Three.js, a higher-level library designed to simplify WebGL programming (https://threejs.org). Carina adapts many 3D accelerated graphics techniques from WebGL to 2D space for graph visualization. In particular, we use level-of-detail, buffer geometry, and particle systems for fast rendering of nodes and edges.

Visual Scalability on Real Data. To better understand WebGL's rendering scalability, we tested Carina with graphs of varying sizes, on a MacBook Pro laptop (2015 version, i7-4870HQ, 16GB RAM). For the YEAST dataset with 2361 nodes and 7182 edges, user interactions such as panning, zooming and dragging nodes, with a force-directed graph layout algorithm running in the background, achieved a smooth frame rate of 60FPS. The frame rate only starts to drop below 30FPS when the graph size exceeds 20k nodes and 56k edges. We note that when analyzing real-world power-law graphs, we typically would not want to visualize the entire graph (which shows up as a "hairball"). We conducted the above experiment to understand the technological limits.

Cross-platform Integration and Deployment. We design Carina with platform portability in mind, hence our decision to use latest web browser technologies. Carina can be deployed as a desktop application that runs on all popular operating systems (Linux, Windows, Mac), via the Electron framework based on the Chrome browser and Node.js (http://electron.atom.io/). Electron packages Carina as self-contained binaries for all platforms without assumptions of the user's run-time environment. Electron also grants Carina native I/O and high performance Inter-Process Communication (IPC) capabilities. Carina can also be used in any modern browsers, allowing it to run on mobile devices and desktops.

Million-node Graph Data Storage and Exploration. Different from most graph visualization tools, Carina does not store the full graph in RAM; only the visualized subgraph is kept in memory, allowing Carina to work with graphs with up to 69M edges [4]. Currently, Carina stores a million-node graph in an out-of-core SQLite database. We chose SQLite for its simplicity, ease of integration, and scalability for up to tens of millions of edges. The user can select and visualize sub-graphs through techniques such as node search (as in Fig.) or based on graph features (e.g., computed measures like PageRank scores). SQLite can induce subgraphs quickly. For example, it takes only 120ms to induce a 2000-node subgraph (9867 edges) out of the 69M edge LiveJournal graph.

3. CONCLUSIONS & ONGOING WORK

We are designing and developing Carina, a scalable, interactive visualization system for million-node graph exploration. Using latest web browser technologies, Carina offers multiple advantages over existing graph visualization tools, such as fast graph rendering (via WebGL), easy cross-platform deployment (via Electron), and scalable data storage and exploration of large graphs with up to 69M edges on commodity machines. We plan to conduct lab studies to evaluate the usability of Carina, and work with real domain users, such as security analysts at Symantec, to use Carina to help uncover company insider threats lurking in large email graphs and computer communication networks. We believe Carina provides a new, scalable way for practitioners and researchers to explore and visualize large graph datasets.

Acknowledgement

We thank Andrei Kashcha for the WebGL inspiration for graphs. This work is supported by NSF IIS-1563816, 1217559, TWC-1526254.

4. REFERENCES

[1] M. Bastian, S. Heymann, M. Jacomy, et al. Gephi: an open source software for exploring and manipulating networks. *ICWSM*, 8:361–362, 2009.

[2] D. H. Chau, A. Kittur, J. I. Hong, and C. Faloutsos. Apolo: making sense of large network data by combining rich user interaction and machine learning. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 167–176. ACM, 2011.

[3] DARPA. *DARPA-BAA-10-50: Graph Understanding and Analysis for Rapid Detection - Deployed on the Ground (GUARD DOG)*, 2010 (accessed Jan 7, 2017).

[4] J. Leskovec and A. Krevl. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data, June 2014.

[5] P. Shannon, A. Markiel, O. Ozier, N. S. Baliga, J. T. Wang, D. Ramage, N. Amin, B. Schwikowski, and T. Ideker. Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome research*, 13(11):2498–2504, 2003.