AdaptiveNav: Discovering Locally Interesting and Surprising Nodes in Large Graphs

Robert Pienta, Zhiyuan Lin, Minsuk Kahng, Jilles Vreeken
Partha P. Talukdar, James Abello, Ganesh Parameswaran, Duen Horng (Polo) Chau

Abstract—Visualization is a powerful paradigm for exploratory data analysis. Visualizing large graphs, however, often results in a meaningless hairball. In this paper, we propose a different approach that helps the user adaptively explore large million-node graphs from a local perspective. For nodes that the user investigates, we propose to only show the neighbors with the most subjectively interesting neighborhoods. We contribute novel ideas to measure this interestingness in terms of how surprising a neighborhood is given the background distribution, as well as how well it fits the nodes the user chose to explore. We are currently designing and developing AdaptiveNav, a fast and scalable method for visually exploring large graphs. By implementing our above ideas, it allows users to look into the forest through its trees.

Index Terms—Graph Visualization, Data Exploration, Adaptivity, Serendipity

1 INTRODUCTION

Visualization is perhaps the most natural approach to exploratory data analysis. Under the right visualization, finding patterns, deciding what is interesting, what is not, and what to investigate next are all easy tasks – in a sense the answers ‘jump to us’ as our brains are highly specialized for analyzing complex visual data. Visualizing large graphs in an intuitive and informative manner has proven to be rather difficult. Even using advanced layout techniques plotting a graph typically leads to a useless “hairball” from which nothing can be deduced [4, 5]. Instead of plotting the whole graph, visualizing only part of the graph seems more promising [7, 2]. Doing this naively leads to the same problem. Because real world graphs are often scale free (follow a power law degree distribution [3]), even a single hop expansion from a selected node can be visually overwhelming.

We take a different approach. We propose to adaptively explore large graphs from a local perspective. That is, starting from an initially selected node – e.g., explicitly queried by the user, or proposed by algorithms [1, 6] – we propose to only show the most interesting neighbors. We identify these by their subjective interestingness, by how surprising their neighborhood distribution is (e.g., do neighbors’ degree distributions follow a power law like when considering all nodes?), as well as by how similar this distribution is to those of the nodes the user already visited during the exploration. By only showing those parts of the graph that are most informative to the user, we keep the view clean.

We call our adaptive approach AdaptiveNav — our idea is a significant addition to existing works that aim to recommend individual nodes to users; instead, we steer users towards local regions of the graphs that match best with their current browsing interests, helping them better understand and visualize the graph at the same time. By being adaptive, AdaptiveNav allows every user to explore those facets of the graph that are most interesting to them.

\[ s_i = \sum_{f_j \in A} \lambda_j D_{JS}(L_{i,j}||G_j), \]  

where \( L_{i,j} \) and \( G_j \) are the local and global distributions of node-feature \( f_j \) and \( \lambda_j \) is a feature weight.
The subject of interest rankings arise from the divergence between the local feature distributions and the user’s current profile, which represents their exploration, in node-features, up to that point. We track the user’s behavior and record a user profile as they explore their data. Each clicked node offers valuable details into the types of nodes in which the user is interested. This forms distributions $U_j$ for each feature $f_j$. To rank the user’s interest in the undisplayed neighbors of node $v_k$ we follow a similar approach as Equation 1.

In summary, whenever a user selects a node to explore, we rank its neighbors based on surprise and subjective interestingness. For each of the neighbors, we compute surprise and interest scores for each feature and aggregate them based on feature weights $\lambda_j$. We blend those scores, and finally, nodes with $k$ highest scores will be returned.

### 2.2 Components

AdaptiveNav’s user interface as shown in Figure 2 has four key elements: The first main area is the **Table View** (1), a conventional table showing the currently displayed nodes and some of their features. This provides sortable node-level information. The central area is the **Graph View** (2). It is an interactive force-directed graph layout that demonstrates the structure and relationships among nodes as the user explores. Coloring the nodes is used to encode the surprise and interest of the neighbors based on the user’s current exploration. The Neighborhood Summary (3) allows a user to investigate the feature distributions of its currently undisplayed neighbors. It presents the user with feature heat maps that summarize the distributions of hidden nodes, which can be clicked and turn into histograms. This lets a user quickly select new nodes based on their feature values and get a quick summary of their currently undisplayed neighbors. As a user explores, we construct and display a summary profile of the important features they have covered in the **User Profile** view (4). The user profile view suggests high-level browsing behavior to the user; allows for better understanding of where the user-interest ranking comes from; and allows them to adjust if they want to ignore certain features in the interest ranking.

### 3 Scenario

To illustrate how AdaptiveNav works in practice, consider our user Susan who is looking for interesting movies to watch (see Figure 3), by exploring a Rotten Tomatoes movie similarity graph with 17k movies. In this graph, an edge connects two movie nodes if users of Rotten Tomatoes voted them as similar. Susan has watched Miss Congeniality, a crime-comedy that stars Sandra Bullock as an FBI agent who thwarts terrorist efforts by going undercover, turning her rude unflattering self into a glamorous beauty queen. AdaptiveNav also suggests a few movies that are interesting and surprising to her.

Matching Susan’s interest, AdaptiveNav suggests the Big Mommas House series, which also have undercover plots and are interestingly like Miss Congeniality. They both share low critics scores, but high audience scores. To Susan’s surprise, AdaptiveNav suggests Pretty Woman, which is quite different (thus surprising) — a romantic-comedy that has both scores from the critics and the audience. But, there is still more subtle similarity (thus still drawing Susan’s interest); both films share a Cinderella-like storyline, which explains why the two movies are connected in the graph: Sandra Bullock goes from a rude agent to a beauty queen; in Pretty Woman, Julia Roberts goes from a prostitute to a fair lady. In fact, Pretty Woman is a classic, exemplar romantic-comedy; many movies follow similar story lines (e.g., Maid in Manhattan). Thus, Pretty Woman has very high degree in the graph, unlike Miss Congeniality which is a niche genre; this also contributes to Pretty Woman’s surprisingness.

Through Pretty Woman, AdaptiveNav again pleasantly surprises Susan with Oceans Eleven, which also stars Julia Roberts, and is in a rather different light hearted crime or heist genre, introducing Susan to other very similar movies like Oceans Twelve and The Italian Job.

### 4 Conclusion

In this work, we describe AdaptiveNav, an integrated approach that combines visualization and computational techniques to help the user performs adaptive exploration of large graphs. AdaptiveNav overcomes many issues commonly encountered when visualizing large graphs. AdaptiveNav shows the user only the most subjectively interesting material as they explore. We do this by ranking the nodes by surprisingness and subjective interest based on what the user has explored so far. The user interest ranking influences the direction of a user’s exploration by showing the best matches to their current taste in nodes. Despite the old adage, you can see the graph through its nodes.

### References


