Graph Signatures for Visual Analytics

Pak Chung Wong, Harlan Foote, George Chin Jr., Patrick Mackey, and Ken Perrine

Abstract—We present a visual analytics technique to explore graphs using the concept of a data signature. A data signature, in our context, is a multidimensional vector that captures the local topology information surrounding each graph node. Signature vectors extracted from a graph are projected onto a low-dimensional scatterplot through the use of scaling. The resultant scatterplot, which reflects the similarities of the vectors, allows analysts to examine the graph structures and their corresponding real-life interpretations through repeated use of brushing and linking between the two visualizations. The interpretation of the graph structures is based on the outcomes of multiple participatory analysis sessions with intelligence analysts conducted by the authors at the Pacific Northwest National Laboratory. The paper first uses three public domain data sets with either well-known or obvious features to explain the rationale of our design and illustrate its results. More advanced examples are then used in a customized usability study to evaluate the effectiveness and efficiency of our approach. The study results reveal not only the limitations and weaknesses of the traditional approach based solely on graph visualization, but also the advantages and strengths of our signature-guided approach presented in the paper.

Index Terms—Data and knowledge visualization, information visualization, visualization techniques and methodologies, graphs and networks.

1 INTRODUCTION

We are interested in exploring graphs (or networks) in order to better understand the stories that are derived from them. Being able to draw descriptive pictures from the graphs [7], [26] is a critical step in advancing our goal. Interactively visualizing the graphs [1], [2] also represents significant progress. However, when it comes to understanding the meaning of the graphs and implications of their contents, we focus our attention on the connectivity of individual nodes and their associated neighbors. This topology information allows us to categorize the basic graph entities and subsequently forms a basis to carry out a discourse of interrogation on the graphs.

Throughout the paper, we borrow terminology from the “small world” studies by Milgram [19], Watts [32], [33], and Watts and Strogatz [34] to explain our work. We follow Milgram’s assertion [19] that one can assign a numerical value to the distance between any two nodes within a network, measured by the length of the chain of intermediate connections between the two nodes. However, we would like to point out that the visual analytic approach presented in this paper is not restricted to small world network analyses. We will show with examples in Section 5 that our approach can be applied to a variety of graphs, including one that depicts a biological network representing the transcriptional regulation of Escherichia coli (E. coli) [10].

The basic topology of a graph can reveal a great deal about the underlying data. For example, we can classify the graph nodes in Fig. 1a into three groups represented by yellow, green, and blue. The yellow node in the middle can be a leader or organizer of the hierarchy. Beneath the yellow node are the secondary (green) and tertiary (blue) actors of the network. In Fig. 1b, both the cyan and magenta nodes are part of two closed loops (or paths.) However, only the cyan nodes can talk to the other loops.

This paper presents a visual analytic technique that characterizes graph nodes based on their topology, summarizes the topology of individual nodes as signature vectors, and projects the vectors onto a low-dimensional visualization for further exploration. Customized signature design approaches are used to bring out different aspects of a graph. Brushing, linking, and clustering are used extensively to cross-examine different visualizations created by different signatures.

Although not a discussion topic of this paper, the semantics information of a graph, if available, should be used to supplement the topological analyses of the graphs. We will revisit the topic of graph semantics in Section 3.

This paper presents a system prototype, known as Greenland, which implements our new signature-based visual analytics technique and a few other supporting tools to address requirements of different applications. Part of our research effort is to evaluate the efficiency and effectiveness of our signature-based design and implementation. We hypothesize our design expectations and compare them with the outcomes of the study. The results are discussed in the second half of the paper.

2 RELATED WORK

The scope of visual-based graph analysis is profound and enormous. We will first highlight major reference books, conference proceedings, and a literature survey that cover prior work on the topic. More recent results that share similarities with our approach will also be discussed. Readers are encouraged to review the reference sections of these cited literatures.
Di Battista et al. [7] suggest that, as early as the 1960s, computer scientists had already applied graph drawing techniques to assist software development. During the following decades, more researchers joined the community to draw larger and more sophisticated graphs more intelligently and efficiently. Many of their contributions are presented in literature [7], [26]. The Proceedings of the Annual International Symposium on Graph Drawing [9] also showcases the latest and best graph drawing work offered by the community. There are also a number of popular graph drawing tools or libraries that address different areas of the problems. Many of them are public domains. Notable examples are Graphviz [12], Jung [15], Pajek [21], and Tulip [27].

Unlike the static graphics approach championed by the graph drawing community, the information visualization community focuses more on dynamic, interactive techniques to visualize and navigate large and complicated graphs. Books by Card et al. [1] and Chen [2] have detailed discussions on the evolution of graph visualization throughout the last decade. Graph visualization is also a major topic at the Proceedings of the Annual IEEE Symposium on Information Visualization [14]. Herman et al. [13] present a very complete survey of major graph visualization work up until 2000. More recently, a series of graph visualization work reported by Chen and Morris [3], van Ham et al. [29], and Chiricota et al. [4] has steered away from the mainstream depiction [20], [28] and navigation [36] approaches and moved toward the more exploratory direction. Some of this work can be traced back to the classic social network analysis book by Wasserman and Faust [31]. The research presented in this paper shares a similar design philosophy, but with a different design approach based partly on the concept of a data signature [37] that we previously used to analyze very large scientific data sets on a modest desktop computer.

3 Bridging Graph Topology with Visual Knowledge

Our graph visualization research is being applied in the field of social network analysis, where investigators often apply a general graph analysis technique known as link analysis [24]. Through link analysis, investigators graphically draw, lay out, and link various people, facts, locations, events, objects, and data items in hopes of identifying key patterns and insights. Link analysis has been applied in a number of high-profile cases including the search for the District of Columbia (US) snipers [23] and the search for Saddam Hussein during the US invasion of Iraq [18]. One quality of link analysis graphs is that they contain uncertainties—particular objects and relationships may be missing from the graph or their existence may be suspect or hypothetical. Many of the link analysis graphs we encounter have properties of small world [19], [32], [33], [34] graphs, which generally have high degrees of clustering and small average path lengths relative to their number of nodes. Small world graphs are commonly associated with social networks, neural networks, power grids, and Internet traffic.

In our continued efforts to visually analyze graphs with data that contain a high degree of unknown factors and uncertainties, we find that we can learn a lot about a graph from the topology of its entities. More specifically, the local topology surrounding individual nodes can reveal a wealth of hidden knowledge if we know how to seek the information and interpret the results.

Table 1 shows a detailed list of detectable graph features and their interpretations using basic small world [19], [32], [33], [34] terminology. The first four features (shown in blue) require both topology and semantic information—such as the node type or exact instance—of the graphs to carry out the analyses. The other 20 features shown in green...
can mostly be addressed by analyzing the topology of individual nodes and their associated neighbors.

The interpretations of Table 1 emerged from multiple participatory analysis sessions with intelligence analysts conducted by the authors at the Pacific Northwest National Laboratory. The analysts were presented with large, pre-defined link analysis graphs, which contain about 1,000 nodes and links each, printed separately on 36" × 48" posters. They were then asked to identify critical patterns in the graphs. During these sessions, analysts would isolate specific subgraphs and then describe and discuss their interpretations of those subgraphs in terms and concepts that were meaningful to them. They identified concepts, such as leaders, followers, messengers, chains of commands, and points of potential disruption in a graph, which were then associated with specific physical properties or features of the graph. For example, a hierarchical communication structure among a number of person nodes might identify a command structure or nodes with the most active edges (high centrality) or singular or weak links that tie two or more portions of the graph together (bridges) might identify potential points of disruption. Analysts’ interpretations often invoke tacit knowledge, where analysts recognize specific intelligence concepts in a graph without being fully aware of their underlying structural properties. In verbalizing their interpretations, however, analysts were challenged to make their tacit knowledge explicit.

Table 1 maps intelligence concepts that analysts found to be important to standard graph properties and measures that are well-known in graph theory [35] and social network analysis [31]. The significance of these mappings is that they allow us to associate signatures to end-user semantics. Thus, a signature that encapsulates specific graphical properties also denotes specific meanings or interpretations to the analysts. Ultimately, this allows us to raise the level of discourse in a signature-based system to better match the language, thought, and conceptions of intelligence analysts.

Unfortunately, automatically identifying graph features, such as those described in Table 1, is mostly an intractable NP-complete [8] task, which is not recommended in an interactive environment. While there are several excellent heuristic algorithms for solving this kind of NP-complete problem [8], we chose to use a less-theoretical visual analytics approach by applying the data signature concept [37] to extract only a small amount of local topology information and then use these signature vectors to analyze the graph’s global structures.

4 FROM LOCAL SIGNATURES TO GLOBAL PROJECTION

We present the design of signature vectors for both directed and undirected graphs in detail. This is followed by discussions on the scaling, projecting, clustering, and brushing of these vectors in an interactive exploration environment.

4.1 Local Topology Extraction

We first traverse the graph and perform a shallow breadth-first search (BFS) at each node to collect its local topology information. The exact depth of the BFS depends on the graph features (as defined in Table 1) that are involved in the analysis. For example, when we are only curious about how the nodes are related to their immediate neighbors, a depth of one is usually enough to describe the topology of the nodes.

There are cases when users want to go deeper—for example, to find out who has the greatest influence over the entire network, find large circuits or follow a money trail. In general, the depth of the BFS should be proportional to the longest path length of the graph features (as defined in Table 1) that are involved in the analysis.

4.2 Data Signature Vector

We then store the local topology information of each graph node as an individual signature vector that represents the node. In the following discussion, we further reduce the size of these signature vectors by summing up the fan-in/fan-out information according to the number of steps from the root of the tree. This basic signature design serves only as a simple illustration to prove the concept of our approach. The implementation can be extended to capture more complex structural features for more detailed analyses than we present in this paper. Here, we use an undirected and a directed graph to illustrate our design.

4.2.1 Undirected Graph Signature

Fig. 2 shows an undirected graph with 12 nodes. The cyan node in the middle is the root of the BFS for this example. The node labels indicate the steps required to travel from the root to those nodes. There are three "1," three "2," three "3," and two "4" labels in Fig. 2. The node counters in distance order become the foundation of our signature design. In this case, a 4-degree signature of the cyan node becomes [3, 3, 3, 2]. Notice that the contents of the vectors are the underlined counters shown earlier in this paragraph. In short, the signature of a d-degree undirected graph node can be defined as a vector \( (n_1, n_2, \ldots, n_d) \), where \( n_i \) is the number of the nodes at distance \( i \) from the node.

Because the recursive BFS returns at nodes that reach the maximum depth or have been previously visited, certain ambiguities are unavoidable in some cases. For example, the link between the two "3" nodes on the right side of Fig. 2 is a blind spot because its presence does not contribute to the outcome of the signature that represents the cyan node.

4.2.2 Directed Graph Signature

For a directed graph like the one shown in Fig. 3, we follow the same BFS approach to traverse the graph. However, this
connected to two blue graph nodes in Fig. 1a.

correspond to the two green graph nodes connected to three clusters. One cluster has two green scatter nodes that are still separated.

The three green scatter nodes in Fig. 4 are very close to each other but are identical because one is a part of a four-node loop and the other blue node cluster has two scatter nodes that correspond to the blue graph nodes with one sibling only in Fig. 1a. Because we used a 3-degree signature vector to represent the topology of the nodes in Fig. 1a and Fig. 1b together, the close-loop structures in Fig. 1b break up instantly, and the local topology of the magenta nodes in Fig. 1b becomes somewhat similar to both blue and green graph nodes in Fig. 1a. That is the reason behind the patterns of the seven magenta scatter nodes that spread out between the green and blue clusters in Fig. 4.

Similarly, there are two blue node clusters in Fig. 4. One has six scatter nodes that correspond to the blue graph nodes with two siblings in Fig. 1a. The other blue node cluster has two scatter nodes that correspond to the blue graph nodes with one sibling only in Fig. 1a.

Because we used a 3-degree signature vector to represent the topology of the nodes in Fig. 1a and Fig. 1b, the close-loop structures in Fig. 1b break up instantly, and the local topology of the magenta nodes in Fig. 1b becomes somewhat similar to both blue and green graph nodes in Fig. 1a. That is the reason behind the patterns of the seven magenta scatter nodes that spread out between the green and blue clusters in Fig. 4.

4.4 Graph Exploration by Brushing and Linking

Depending on the problem at hand, there are different ways to apply the above graph and scatterplot pairs to explore a graph. Here, we suggest two common scenarios.

Analysts can 1) identify an interesting structure such as a hierarchy on the graph visualization, 2) brush [5] the graph nodes that are associated to the structure, and then 3) link [5] the brushed nodes to the scatterplot visualization. Because the scatterplot is scaled to show the similarities of the graph nodes, all the nearby neighbors surrounding the linked scatteredots that are corresponding to the brushed graph nodes in the scatterplot will share similar topology.

If the graph drawing is too busy to identify any visible structures, analysts can focus on the scatterplot first by brushing clusters such as high-density focused areas, low-density dispersed areas, outliers, or anything in between in different parts of the scatterplot. The brushed scatteredots can then be linked to the corresponding graph nodes in the graph visualization by, for example, highlighting the structures and bringing them to the foreground of the otherwise messy display. Analysts can now study the relationships among the real-life data entities represented by these graph nodes.

4.5 Graph Exploration by Scatterplot Clustering

A slightly different approach to using a scaling scatterplot to supplement the analysis of a graph is to subdivide the scatterplot into a number of clusters using statistical methods such as K-mean [22] and then study the interrelationship between the clusters and the corresponding features on the graph. Perhaps a stronger reason to cluster the scatterplot is to study the demographics of individual clusters simultaneously by 1) analyzing the overall distribution of the graph structures and 2) annotating the clusters for further investigation.

K-mean has a relatively effective algorithm that requires little computational resources to operate. We were able to process a scatterplot with about 10,000 scatteredots in interactive time using a C++ compiled program running on a modest notebook PC.

Determining the right values of “k” (or the number of clusters) for any K-mean computations is itself a part of the graph exploration experience. A smaller k gives a coarser division and, thus, an overview, whereas a bigger k gives a finer division and, thus, the details. For example, from a point far, far away from earth, there is only one globe (k = 1). If we move closer, we see oceans and lands (k = 2). Moving closer,
we will see regular lands versus lands covered by snow (k = 3) (i.e., while we find more details on land, the oceans remain the same). As we move closer and closer to the earth, we will progressively see cities, mountains, forests, and more, and the number of k will keep increasing. We argue that each value of k gives different meanings to the clusters and, thus, there is no one standard to decide the optimal values of k.

Fig. 5a, Fig. 5b, and Fig. 5c show three K-mean examples where k = 2, 3, and 4 using the scatterplot in Fig. 4. In Fig. 5a, where k = 2, nodes on the right cluster (in magenta) have at most two neighbors, whereas nodes on the left cluster (in green) have more. The magenta cluster in Fig. 5a is further divided into two smaller clusters in Fig. 5b, where k = 3. In Fig. 5b, the nodes in the blue cluster are all leaf nodes and the ones in the magenta cluster are all part of a closed loop in Fig. 1. Finally, in Fig. 5c, where k = 4, the left side of Fig. 5a is divided into two smaller clusters that separate the yellow node, which is the root of the hierarchy in Fig. 1, and the others.

5 DEMONSTRATIONS USING PUBLIC DOMAIN GRAPHS

We use three benchmark graphs previously used in the graph drawing contests at the Proceedings of the International Symposium on Graph Drawing 96 (GD96) [11] and 2003 (GD03) [10] to demonstrate the versatility of our visual analytics approach. Both the first (telephone connections) and second (fragment of WWW) graphs are widely considered by domain experts as classic small world networks. The third graph is a biological network that reflects scientific fact rather than social or psychological phenomena.

We intentionally select well-known public domain graphs with either obvious structures or hidden structures that are well-known to the community to show the versatility of our signature design. More complicated graph examples will be presented later in the usability study discussion in Section 7.

5.1 An Undirected Phone Call Graph

The first example (GD96B) is an undirected graph that contains 111 nodes and 193 links. The file is extracted from a large telephone-call database. We discard the direction of the links and make it an undirected graph in our demonstration.

Fig. 6 shows a forced-directed representation of GD96B [11] using the Kamada and Kawai [16] layout technique. The graph is relatively small but rich in structural features for our demonstration purposes. For example, there are "closed-paths" in the upper right and lower left corners and an "organizer" surrounded by a large number of "followers" in the middle of the graph. There is also a "hierarchy" in the lower left side as highlighted in the figure. Overall, the force-directed drawing manages to reveal most of the major features as we expected, except one that we will discuss later.

Our first step is to create data signatures that characterize the local topology of individual graph nodes. In this particular example, we choose to create 3-degree signatures because the size of this graph is relatively small, and we only want to pick up the local topology of individual nodes.

The next step is to normalize these multidimensional signature vectors and apply a K-mean [22] clustering process to group them into nine clusters. (As we mentioned earlier, the value of k = 9 was determined after multiple experiments to best annotate our graph.) We then use a classical MDS [6] to project the vectors onto a two-dimensional scatterplot. Each cluster receives a unique color and ID as shown in Fig. 7. The blue and cyan annotations in Fig. 7 are human add-ons to explain the results.

Finally, we map the cluster colors of individual scatter nodes in Fig. 7 back to the corresponding graph nodes in
Fig. 5; the result is shown in Fig. 8. Seven sample signature vectors are selected and depicted as bar graphs in Fig. 8 to show the differences among the signatures of each cluster. The construction of the bar graphs shown in Fig. 8 was previously explained in Section 4.2.1.

We notice that all the major features in Fig. 6 are properly clustered by the K-mean process in Fig. 7 and Fig. 8. For example, the hierarchy is shown in green (cluster 4) and light blue (cluster 3); the closed loops are in magenta (cluster 8), beige (cluster 2), and brown (cluster 9); the followers are in light purple (cluster 6) and yellow (cluster 5); and, finally, the organizers are in red (cluster 1). The result is a strong indication that our signature vector design for an undirected graph is performing as we expected. In other words, the signature vectors capture the right kind of topology information that correctly characterizes individual nodes in the scaling process.

A surprising discovery in this example is the detection of two organizers (instead of one as suggested in Fig. 6) among a massive number of followers in the center of the graph in Fig. 8. The previously missing leader (node 9) is now clearly shown in red in both Fig. 7 and Fig. 8.

One may argue that the missing leader (node 9) may not be missing after all if we use a larger (than Fig. 6 and Fig. 8) figure. But, in reality, even trained professionals make mistakes and overlook noisy clues like this one due to a variety of factors including the so-called “analyst fatigue.”

5.2 A Directed World Wide Web Graph

The second example (GD96D) [11] is a directed graph that contains 180 nodes and 229 links. The graph represents a fragment of the AT&T Web site that contains several nucleus nodes and large hierarchies underneath them.

We follow the same analysis steps in Section 5.1 to

1. generate 3-degree signature vectors to represent local topology of individual nodes,
2. group the signature vectors into 10 clusters that share strong similarities,
3. project the signature vectors onto a 2D scatterplot as shown in Fig. 9, and
4. draw the graph and use corresponding cluster colors to paint the graph nodes in Fig. 10.

Once again, we choose to obtain 10 clusters in Step 2 so that nodes with different topology are clearly annotated in the example. The number of clusters was determined after multiple experiments.

An important difference between this example and the last one is the length of the signature vectors. The 3-degree signature vectors created for this directed graph have six numbers, which are double the size of the signature vectors of the last example because they store both the fan-out and fan-in information separately. Fig. 10 shows several examples of these signature vectors with fan-out and fan-in information represented in blue and red. The construction of the bar graphs shown in Fig. 10 was previously explained in Section 4.2.2.
A linking analysis between Fig. 9 and Fig. 10 shows that the nucleus nodes of the graph (or roots of the hierarchies) are all pushed to cluster 9 (red) in the upper left corner of the scatterplot in Fig. 9. On the other hand, the leaves of the hierarchies are gathered in clusters 6 (green), 3 (light blue), and 7 (dark blue) located in the lower right corner in Fig. 9. All the rest of the mid-level nodes within the hierarchies are correctly spread out between these two extreme corners.

The result of this example shows that our signature vector design is strong enough to characterize the local topology of a directed graph for further analyses such as the scaling and clustering of graph entities.

5.3 A Directed Biological Graph

The third and last example (GD03A) is a directed graph that contains 423 nodes and 578 links. It represents transcriptional regulation of *E. coli*. The original goal of this graph drawing contest [10] question is to highlight the frequent occurrence of the so-called network motifs [10] (or small isolated subgraphs) within the biological graph. We specifically select a biological network example because we want to show the versatility of our data signature design in representing different types of graphs.
We treat the *E. coli* network as a directed graph and follow the same procedures described earlier in Section 5.2 to process the graph. After the signature vectors are computed and scaled, we have an MDS scatterplot in Fig. 11 and a force-directed graph in Fig. 12. There are a lot of very visible structural features identified by the unique cluster colors in both figures. But we want to focus on locating the so-called network motifs within the graph.

The K-mean process highlights the similarities of signature vectors corresponding to the network motifs and gathers them into clusters 8 (orange) and 10 (brown) in Fig. 11. The subgraphs that contain one to two nodes are collected in cluster 8; the others are collected in cluster 10. This example is yet another strong indication that our signature-based visual analytics approach can handle graphs that go beyond small world networks.

6 Computational Performance

We have implemented the visual analytics technique presented here on a system prototype known as Greenland. The system consists of a friendly front-end implemented in Java and a server back-end implemented in C++. Among the most time-consuming processes in the back-end, besides the force-directed layout routine, are the signature construction and the MDS processes.

To show that our visual analytics approach is practical in an interactive environment, we have conducted a performance study using the three demo data sets (GD96B, GD96D, GD03A) on a Dell dual-3.2GHz Xeon running Windows XP Pro. We also include the performance results.
of two larger graphs (Beethoven and Diabetes) harvested from the public domain [25].

Table 2 shows the results (in wall-clock seconds) of the 3-degree signature creation and MDS processes using both directed and undirected versions of the graphs. The nearly-zero MDS column is included in the table for completeness. While we record a timing difference between the creations of directed and undirected graph signatures for GD96B, the results of the other four graphs in Table 2 are identical.

As it turns out, the time spent on MDS is totally insignificant in our performance study. Even though the signature-creation task is slightly more expensive, the figures in Table 2 show that they are all acceptable (less than one wall-clock second) in an interactive environment for graphs with a few thousands nodes and links.

7 Usability Evaluation

We conducted a usability evaluation of Greenland with participants from PNNL. For the evaluation, we derived four graph-based analysis problems that participants were to solve using 1) only the general graphing capabilities available in Greenland (e.g., graph layout, panning, zooming, moving nodes and links, resizing icons, and resizing arrowheads) and 2) the general graph viewing capabilities plus the scatterplot and clustering features in Greenland. The main objective of the evaluation was to assess the value of applying and viewing topological information (as presented through scatterplots and clusters) in graph-based analyses. The analysis problems were deliberately designed to represent scenarios where graph topology was a fundamental or key aspect of the problem. Our underlying hypothesis for the evaluation was that access to topological information allows users to conduct specific kinds of graph analyses more accurately, more quickly, and with greater satisfaction.

7.1 Participants, Setup, and Procedures

Sixteen participants from different departments within PNNL participated in the usability evaluation. The participants were junior researchers and developers, who had no or very limited prior knowledge of and experience with Greenland. Participants took part in individual sessions that lasted 20 to 30 minutes. Evaluations were conducted in a computer lab using a Windows desktop computer, where the Greenland software had been installed. The computer was attached to two 20-inch flat panel monitors. Participants interacted with the Greenland software primarily through mouse interactions. During each participant session, one evaluator would explain the analysis problems while another would track the time required to complete each problem as well as collect general observations.

At the beginning of each session, an evaluator would demonstrate the general use of the Greenland software to the participant—specifically showing the participant how to use the mouse to explore the graph, how to create a scatterplot, and how to cluster the scatterplot. The participant was then handed a sheet of paper containing the set of analysis problems. For each problem, an evaluator would describe and elaborate on the problem and answer any questions. When ready, the participant attacked the same problem twice using Greenland on two different graphs or subgraphs of similar complexity—one without and once with the scatterplot and clustering capabilities. Half the participants were tasked to use the general graphing capabilities (without scatterplot and clustering) first to solve the problems, while the other half were tasked to use the scatterplot and clustering capabilities first. When participants were tasked to solve a problem using scatterplots and clustering, the signature depth and number of clusters were consistently set by an evaluator to ensure that all participants were working from the same visualizations. Varying signature depths and clustering numbers would have yielded visualizations with different features and levels of complexity and, thus, would have made reasonable comparisons across participants difficult to achieve. For each task, the participant was timed, observations were gathered, and the participant’s solution or answer was collected.

7.2 Analysis Problems

The four problems given to participants focused on specific structural analysis tasks or investigations that scientists and analysts may commonly perform in their regular work. We recognize that scientists and analysts look for specific structural patterns or features in a semantic graph, and the problems we derived emphasized some of these fundamental patterns or features. In the testing of each problem, we compared the accuracy of the results and the time required to derive an answer using the two methods. Participants also provided a subjective rating on a 5-point scale to indicate their satisfaction of using the scatterplot and clustering features over the use of general graphing capabilities alone. Below, we present each graph analysis problem, its associated tests, its real-world analogy, its objective, and its evaluation results from participant testing.

7.2.1 Problem 1—Find Isolated Nodes that Connect Two Subgraphs Together: We Call These Isolated Nodes Liaison or Middleman Nodes

Test A. Given a connected graph (Fig. 13a), the participant was to find all liaison nodes that had two links, each of which connected to adjacent nodes that were part of high-density areas of the graph. The participant was to explore
the graph using the general graphing capabilities available in Greenland.

Test B. Given a second connected graph (Fig. 13b), the participant was to again find the liaison nodes but was required to use the scatterplot and clustering features in Greenland along with its general graphing capabilities.

Real-world analogy. In intelligence analysis, an analyst might want to identify and suppress a liaison node to disrupt collaboration between two groups or to stop a particular scenario from happening. A chemistry researcher might wish to remove a liaison node to stop a chemical reaction from occurring.

Objective. We hypothesized that participants would be able to identify liaison nodes more accurately and quickly using Greenland’s scatterplot and clustering features. Using the graph alone, finding nodes with only two links would be time-consuming and tedious as the user would have to examine each individual node in the graph.

For complex graphs, the user would likely have to pan, zoom, and adjust the graph to be able to see all nodes and links. Using the scatterplot and clustering, however, all liaison nodes should cluster together in the scatterplot. Fig. 13c shows a scatterplot for the graph of Test B with the three liaison nodes surrounded by a white oval box. The scatterplot was generated with a signature depth of two and no clustering.

Evaluation Results. In terms of accuracy, all 16 participants (100 percent) were accurate in finding the three liaison nodes.
nodes in both tests. In terms of required time, participants completed Test A in the average time of 42.8 seconds and Test B in the average time of 30.1 seconds. Thus, participants improved their performance by 12.7 seconds on average using the scatterplot and clustering. In terms of user satisfaction, participants were asked to rate their opinions as to whether using the scatterplot and clustering features of Greenland was more effective in finding liaison nodes compared to using general graphing capabilities alone. Using a 5-point scale (1—strongly disagree, 2—mildly disagree, 3—neither agree nor disagree, 4—mildly agree, 5—strongly agree), participants’ average rating for this question was 3.0, which indicates a neutral experience with the scatterplot and clustering.

7.2.2 Problem 2—For a Particular Node with High Degree, Find Adjacent Nodes that Are Connected

**Test A.** Given a connected graph (Fig. 13d), the participant was to find nodes that have connections to other nodes in the round upper node collection other than the central node. The participant was to explore the graph using the general graphing capabilities available in Greenland.

**Test B.** Given the same connected graph (Fig. 13d), the participant was to find nodes that have connections to other nodes in the round lower node collection other than the central node. The participant was required to use the scatterplot and clustering features in Greenland along with its general graphing capabilities.

**Real-world analogy.** The graph might represent communications among a leader and his or her followers. Connections among adjacent nodes would indicate internal communications occurring among the followers. In a crime scenario, the graph might link various suspects to a crime scene. Connections among adjacent nodes would highlight associations that might imply that certain suspects were conspiring in the crime.

**Objective.** We hypothesized that participants would be able to identify adjacent node connections more accurately and quickly using Greenland’s scatterplot and clustering features. Using the graph alone, links between adjacent nodes may be difficult to detect among the many links emanating from the central node. Using the scatterplot and clustering, however, adjacent nodes with connections should group into different clusters than adjacent nodes with no connections. Fig. 13e shows a scatterplot for the graph in Test B with adjacent nodes with connections colored in white. Four of these nodes appear in the scatterplot because the upper and lower node collections each contain two adjacent nodes with connections. The scatterplot was generated with a signature depth of one and four clusters. When clusters are brushed back onto the graph, adjacent nodes with connections should show up in a different color than those without. Fig. 13f shows the graph with nodes brushed according to their clusters. As with the scatterplot, adjacent nodes with connections are colored white in the brushed graph.

**Evaluation Results.** In terms of accuracy for Test A, 15 of 16 participants (93.8 percent) were able to find the two connected adjacent nodes using the graph alone, while one of 16 participants (6.3 percent) was not able to locate either of the connected adjacent nodes. Overall, participants found 30 of 32 total nodes (93.8 percent) that the participants combined were to find (16 participants × two nodes) for Test A. For Test B, all 16 participants (100 percent) were able to find the two connected nodes using the scatterplot and clustering. In terms of required time, participants completed Test A in the average time of 56.6 seconds and Test B in the average time of 7.6 seconds. Thus, participants improved their performance by an impressive 49 seconds on average using the scatterplot and clustering. In terms of user satisfaction, participants were asked to rate their opinions as to whether using the scatterplot and clustering features of Greenland was more effective in finding adjacent node connections compared to using general graphing capabilities alone. Participants’ average rating for this question was 4.6, which indicates a positive experience with the scatterplot and clustering.

7.2.3 Problem 3—For a Particular Node with High Degree, Count the Number of Ingoing Links

**Test A.** Given a connected graph (Fig. 13g), the participant was to count the number of ingoing links in the upper-left node collection. The participant was to explore the graph using the general graphing capabilities available in Greenland.

**Test B.** Given the same connected graph (Fig. 13g), the participant was to count the number of ingoing links in the lower-left node collection. The participant was required to use the scatterplot and clustering features in Greenland along with its general graphing capabilities.

**Real-world analogy.** Inflows to a node may represent many different kinds of real-world activities, such as the number of people who have traveled to a particular location, suspects that had visited the scene of a crime, troops under the command of a military officer, and deposits that were made into a banking account.

**Objective.** We hypothesized that participants would be able to count the ingoing links more accurately and quickly using Greenland’s scatterplot and clustering features. Using the graph alone, discriminating the number and directions of links may be time-consuming and tedious. The user may have to pan, zoom, and adjust the graph to be able to see all nodes and links. Using the scatterplot and clustering, however, adjacent nodes connected to ingoing links should group into different clusters than adjacent nodes connected to outgoing links. Fig. 13h shows a scatterplot for the graph of Test B with adjacent nodes attached to ingoing links colored in white. The scatterplot was generated with a signature depth of one and four clusters. When clusters are brushed back onto the graph, adjacent nodes attached to incoming links should show up in a different color than those to outgoing links. Fig. 13i shows the graph with nodes brushed according to their clusters. As with the scatterplot, adjacent nodes attached to ingoing links are colored white in the brushed graph.

**Evaluation Results.** In terms of accuracy for Test A, 11 of 16 participants (68.8 percent) counted all 13 of the ingoing links; four of 16 (25 percent) counted 12 of the 13 links; and 1 of 16 (6.3 percent) counted 10 of the 13 links. Overall, participants found 201 of 208 total links (96.6 percent) that
the participants combined were to find (16 participants × 13 links) for Test A. For Test B, all 16 participants (100 percent) were able to count the 13 ingoing links. In terms of required time, participants completed Test A in the average time of 46.3 seconds and Test B in the average time of 16.2 seconds. Thus, participants improved their performance by an impressive 30.1 seconds on average using the scatterplot and clustering. In terms of user satisfaction, participants were asked to rate their opinions as to whether using the scatterplot and clustering features of Greenland was more effective in finding the most influential node compared to using general graphing capabilities alone. Participants’ average rating for this question was 4.3, which indicates a positive experience with the scatterplot and clustering.

7.2.4 Problem 4—Find the Node with the Greatest Influence over the Graph

Test A. Given a connected graph (Fig. 13j), the participant was to identify the node that could reach the greatest number of other nodes in the graph. The participant was to explore the graph using the general graphing capabilities available in Greenland.

Test B. Given a second connected graph (Fig. 13k), the participant was to again identify the node that could reach the greatest number of other nodes in the graph but was required to use the scatterplot and clustering features in Greenland along with its general graphing capabilities.

Real-world analogy. In a social network, the node that is able to reach the greatest number of other nodes may indicate the most influential and powerful leader of a group or organization. In intelligence analysis, the most influential node, whether it be a person, resource, or event, would be a logical target for disruption to prevent a particular scenario from occurring.

Objective. We hypothesized that participants would be able to identify the most influential node more accurately and quickly using Greenland’s scatterplot and clustering features. Using the graph alone, tracking multiple paths across different nodes and links may be very time-consuming and tedious. The user may have to pan, zoom, and adjust the graph to see all nodes and links. Using the scatterplot and clustering, however, the most influential node is likely to have a unique signature with high vector values and, thus, sit in an extreme and/or isolated position in the scatterplot. Fig. 13j shows a scatterplot for the graph of Test B with the most influential node circled. When selecting this node on the scatterplot, the corresponding node back in the original graph will highlight to indicate the most influential node. Fig. 13m shows the original graph with the most influential node circled. The scatterplot of Fig. 13j was generated with a signature depth of one and no clustering.

Evaluation Results. In terms of accuracy, only one of 16 participants (6.3 percent) was able to find the most influential node for Test A, while 15 of 16 participants (93.8 percent) picked an incorrect node. For Test B, 13 of 16 participants (81.3 percent) were able to find the most influential node, while three of 16 participants (18.8 percent) picked an incorrect node. In terms of required time, participants completed Test A in the average time of 92.6 seconds and Test B in the average time of 81.0 seconds. Thus, participants improved their performance by 11.6 seconds on average using the scatterplot and clustering. In terms of user satisfaction, participants were asked to rate their opinions as to whether using the scatterplot and clustering features of Greenland was more effective in finding the most influential node compared to using general graphing capabilities alone. Participants’ average rating for this question was 4.4, which indicates a positive experience with the scatterplot and clustering.

7.3 Summary of Evaluation Results

Our evaluation results are summarized in Fig. 14a, Fig. 14b, and Fig. 14c. With respect to accuracy, Fig. 14a shows that participants improved their accuracy in two of the four analysis problems by using the scatterplot and clustering. The accuracy improvement was particularly striking in finding the most influential node for Problem 4. In developing the test graphs for the study, we sought to limit the complexities of the graphs to the point where we felt that a majority of the participants could solve the problem using the graph alone. The test graphs in the context of Problem 4, however, proved to be more complex than we envisioned as most of the participants failed to correctly solve the analysis problem using the graph alone.

An important quality to note of the test graphs we developed and presented to participants was that, as the complexities of the graphs increased, participants had greater difficulty finding correct solutions without the scatterplot and clustering features due to the sheer number of nodes and edges that participants had to examine, traverse, and explore. With very large graphs (e.g., thousands of nodes and tens of thousands of edges), participants generally were not able to find all liaison nodes or the most influential node using the graph alone. With very dense graphs (e.g., hundreds of edges from individual nodes), participants would have had more difficulties in finding all adjacent nodes with connections or counting the number of ingoing links using the graph alone. As such, we generally expected participants’ accuracies to approach 0 percent as graph complexities greatly increased.

With respect to time, Fig. 14b shows that participants were able to reach a solution more quickly using the scatterplot and clustering for all four problems. The speed improvement was particularly dramatic for Problems 2 and 3, where the answers essentially “popped out” on the graph as distinctly colored nodes. Problems 1 and 4 required the participants to examine and explore the scatterplot in greater detail and, thus, the speed improvement was not as impressive. Furthermore, with regard to Problem 4, recall that only one of the 16 participants was able to derive the correct solution using the graph alone. During this task, participants generally took time to manipulate and adjust the graph to better viewpoints of congestion in the graph, but would then visually estimate the most influential node. If we had insisted that participants verify their answers by counting the number of connections to nodes, we expect that the times to solution would have significantly increased.

With the exception of one problem, Fig. 14c shows that participants strongly felt that the scatterplot and clustering features were more effective than using the graph alone to solve the analysis problems. Participants had a more neutral opinion of the scatterplot and clustering features, however,
when applied to Problem 1. In posttest interviews, participants expressed that the clustering did not behave as expected for Problem 1. While the liaison nodes appeared in the same vicinity in the scatterplot (see Fig. 14c), they were not placed into the same cluster. As such, when the clusters of the scatterplot were mapped back onto the original graph, the liaison nodes were not brushed in the same color.

Poststudy analysis of the graph and the node signatures revealed that the liaison nodes had differing signature vectors in the second vector component. This difference was large enough to cause confusion. We tested one approach to addressing the issue by introducing a signature vector weighting function. For the problem of identifying the liaison nodes, the first component of the vector was more relevant than subsequent components. The liaison nodes were successfully clustered after applying a forward weighting to the vectors to emphasize the importance of the initial two-node connection.

Another general observation from our study was that participants extensively applied brushing and linking actions when using the scatterplot and clustering features of Greenland. Participants would brush distinctive portions of the scatterplot such as isolated nodes, specific clusters, or unique patterns to view the corresponding nodes in the main graph. Conversely, participants would also brush critical nodes and portions of the main graph and then look for points and patterns to emerge in the scatterplot. Participants generally understood the scatterplot to be a second view of the graph data and conducted investigations and analysis primarily by examining the connections across the two views using brushing and linking.

8 Discussion
After intensively testing our system prototype with different types of graphs by different end users, we have developed a list of strengths of our design and weaknesses for further improvements.

8.1 Strengths
The concept of a data signature is easy to understand and simple to implement. Even though we have only implemented a very basic version (i.e., sum-up graph connections

![Average Accuracy](image)

![Average Time](image)

![Average User Satisfaction](image)

Fig. 14. Average scores from usability evaluation of four analysis problems considering (a) average accuracy, (b) average time, and (c) average user satisfaction.
according to the distance from the root of a BFS hierarchy), we are able to do a lot of meaningful and productive analyses as presented in this paper.

The concept of a data signature is also extremely versatile for further enhancements. For example, we can easily customize the signature contents by collecting different types of heuristic knowledge when we traverse the tree structure as shown in Fig. 2 and Fig. 3.

We have found that it is very easy to explain the concept of a data signature to end users. They have had no problem relating the concept of graph topology to their real-life graph problems.

8.2 Weaknesses

The current design lacks intelligence beyond topological facts and their implications for a graph. In other words, we cannot harvest deeper knowledge about the semantics of a graph and its entities using the current implementation. For example, we can only identify the followers of all the leaders, but we cannot determine the followers of a particular leader.

The time requirement of a brute-force BFS is still a factor in dealing with large graphs that contain thousands of nodes and links. So far, we have not done enough studies to claim that our approach is appropriate to interactively analyze graphs beyond a few thousands entities.

Other major factors that challenge the design of Greenland include graph sizes and display pixels. While the current Kawada and Kawai implementation is relatively slow for bigger graphs, Section 9 highlights our plan to speed up the drawing process.

9 ONGOING AND FUTURE WORK

To address the scalability issue of Greenland when dealing with larger graphs with hundreds of thousands of nodes, we are in the process of replacing the Kamada and Kawai technique with a much faster one recently developed by Walshaw [30]. The new multilevel graph drawing technique supports a linear algorithm that is an order of magnitude faster than the Kamada and Kawai technique.

Another important enhancement for Greenland is to incorporate the semantics information (if available) of a graph in our signature design. The new additions will substantially extend our capability to do finer analyses than that related to the topology alone.

Finally, we have developed an interactive visualization tool, known as GreenSketch [38], to generate graphs for visual analytics. We plan to use the graph sketching capability powered by GreenSketch to support graph query in Greenland.

10 CONCLUSION

We present a novel visual analytics paradigm and a system to explore a wide variety of directed and undirected graphs using the concept of a data signature. Multiple graph data sets harvested from public domains are used to demonstrate the concept of our approach and explain the results. More advanced graphs are also used in the usability study that shows the superiority of using the graph alongside its signature-based scaling scatterplot. We have shown in all cases that our signature-based paradigm can be a viable option to analyze a graph beyond merely “drawing” and “visualizing.”

ACKNOWLEDGMENTS

This work has been sponsored in part by the National Visualization and Analytics Center® (NVAC®) located at the Pacific Northwest National Laboratory in Richland, Washington. The Pacific Northwest National Laboratory is managed for the US Department of Energy by the Battelle Memorial Institute under Contract DE-AC05-76RL01830.

REFERENCES


Pak Chung Wong received the PhD degree in computer science from the University of New Hampshire in 1997. He is a chief scientist and project manager at the Pacific Northwest National Laboratory in Richland, Washington. He chairs/cochairs the IEEE Symposium on Information Visualization (InfoVis) 2006, the IEEE Symposium on Visual Analytics Science and Technology (VAST) 2006, and the SPIE Conference on Visualization and Data Analysis (VDA) 2006. His current research interests include visual analytics, visualization, bioinformatics, and homeland security.

Harlan Foote received the BS degree in physics from Washington State University in 1966. He is a senior research scientist at the Pacific Northwest National Laboratory, where he has worked on a wide variety of remote sensing projects. His interests include hyperspectral image classification, multiscale image processing for stereo matching, and multisensor data fusion. He is currently working on problems of feature extraction from 3D millimeter wave holographic images.

George Chin Jr. received the PhD degree in computer science from Virginia Tech. He is a chief scientist at the Pacific Northwest National Laboratory in Richland, Washington. His main area of expertise is in human-computer interaction. At PNNL, he has conducted extensive user and work studies with intelligence analysts and scientists from various domains. Other current research interests include visual analytics, computer-supported collaborative work, social networks, scientific workflow, and scientific problem-solving environments.

Patrick Mackey received the BS degree in computer science from Washington State University in 2004. He is a scientist at the Pacific Northwest National Laboratory, where he has worked on multiple visual analytics projects. His research interests include visualization, scientific computation, and computer graphics.

Ken Perrine received the BS degree in computer engineering from Pacific Lutheran University in Tacoma, Washington, in 1998. He is a senior research engineer at the Pacific Northwest National Laboratory, where he has worked on a wide variety of image processing, computer graphics, and scientific visualization projects. His research interests include parallel image processing, video editing and production, and user interface design.

For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/publications/dlib.