TreePlus: Interactive Exploration of Networks with Enhanced Tree Layouts

Bongshin Lee, Cynthia S. Parr, Catherine Plaisant, Benjamin B. Bederson, Vladislav D. Veksler, Wayne D. Gray, and Christopher Kotfila

Abstract—Despite extensive research, it is still difficult to produce effective interactive layouts for large graphs. Dense layout and occlusion make food webs, ontologies, and social networks difficult to understand and interact with. We propose a new interactive Visual Analytics component called TreePlus that is based on a tree-style layout. TreePlus reveals the missing graph structure with visualization and interaction while maintaining good readability. To support exploration of the local structure of the graph and gathering of information from the extensive reading of labels, we use a guiding metaphor of "Plant a seed and watch it grow." It allows users to start with a node and expand the graph as needed, which complements the classic overview techniques that can be effective at (but often limited to) revealing clusters. We describe our design goals, describe the interface, and report on a controlled user study with 28 participants comparing TreePlus with a traditional graph interface for six tasks. In general, the advantage of TreePlus over the traditional interface increased as the density of the displayed data increased. Participants also reported higher levels of confidence in their answers with TreePlus and most of them preferred TreePlus.

Index Terms—Graph visualization, information visualization, navigation techniques, interaction techniques, evaluation/methodology, graphical user interfaces, Piccolo Zoomable User Interface (ZUI) Toolkit.

1 Introduction

Graph visualization is an important component of Visual Analytics that can be used in a variety of applications from engineering to sociology to biology. For example, scientists can discover unexpected influences on ecosystems by studying complex food webs. Intelligence or corporate law analysts can derive insight by scanning evidence of communication between individuals. Financial networks, distribution networks, and gene ontologies may also be represented as graphs and, as such, have been used to discover illegal activities or to make life saving discoveries.

There are several important issues in the design of interactive graph visualizations. One issue is the size of the graph. While a layout algorithm may produce good layouts for graphs of up to several hundred nodes, it may not be able to scale well to several thousand nodes. Only a few systems such as Tulip [2], Gem-3D [10], HDE [20], H3Viewer [30], and NicheWorks [39] can handle large graphs. Second, any interactive visualization requires near real-time performance. However, most useful operations for drawing general graphs have been proven to be NP-complete [9]. Third, even if a system can lay out and display large graphs, the cognitive demands placed on users by the visualization can be overwhelming [38] since the nodes may be very close together or occlude each other, and the links may cross one another. Furthermore, whereas most techniques attempt to show the entire overview of the graph—which can be effective for revealing patterns and clusters—labels are usually ignored.

In contrast, trees can be laid out nicely in a plane in linear time. They are easy to understand and nicely support abstraction and aggregation. For that reason, some researchers have extracted trees from graphs (e.g., spanning tree) and visualized this tree rather than the graph [22].

We propose an interactive graph visualization called TreePlus that enables users to iteratively explore a graph by starting at a node and then incrementally expanding and exploring the graph (Fig. 1). Our approach involves transforming a graph into a tree plus cross links (i.e., the additional links that are not represented by the spanning tree) and using visualization, animation and interaction techniques to reveal the graph structure while preserving readability of the labels. In contrast with the more familiar overview techniques [34], which are effective at (but also limited to) revealing overall structure and the existence of clusters or bridges, our technique addresses the needs of users to explore parts of the graph in detail and rapidly read labels to analyse the meaning of relationships.

2 Related Work

Graph visualization has been studied extensively over the last few decades [12], [13], [22]. The basic graph drawing problem can be defined simply as: given a set of nodes with a set of links, calculate the position of the nodes and the curve to be drawn for each link. However, most classic graph drawing algorithms have not been developed with

- *B. Lee and B.B. Bederson are with the Human-Computer Interaction Lab, Department of Computer Science, University of Maryland, College Park, MD 20742. E-mail: {bongshin, bederson}@cs.umd.edu.*
- *C.S. Parr and C. Plaisant are with the Human-Computer Interaction Lab, University of Maryland, College Park, MD 20742. E-mail: {csparr, plaisant}@cs.umd.edu.*
- *V.D. Veksler, W.D. Gray, and C. Kotfila are with the CogWorks Lab, Cognitive Science Department, Rensselaer Polytechnic Institute, Troy, NY 12180. E-mail: {kotfic, vekslv, grayw}@rpi.edu.*

Manuscript received 2 Dec. 2005; revised 24 Feb. 2006; accepted 7 Mar. 2006; published online 8 Sept. 2006.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number TVCG-0212-1205.

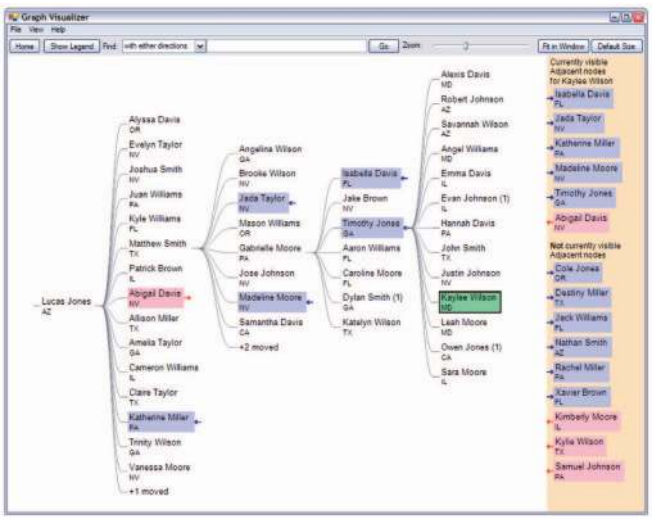

Fig. 1. TreePlus with the low density data set used in the user study. A single click on any node (here, "Kaylee Wilson") highlights adjacent nodes already present in the tree and lists new names in the preview panel on the right. Color indicates the direction of the link. Double-clicking on a node expands the tree by adding new adjacent nodes and moving existing nodes as needed (see video demonstration at http://www.cs.umd.edu/hcil/treeplus).

interaction in mind [22]. For example, force-directed algorithms [13] are very popular because they are simple and easy to understand. However, they are usually slow and produce a different final layout each time the algorithm is invoked, which is disorienting. Furthermore, when they are applied to graphs with labeled nodes, the resulting layouts suffer from severe node occlusions [17].

A number of researchers have tried to visualize graphs as trees. Hao et al. visualized large highly connected hierarchical graphs in a hyperbolic space using an "invisible link" technique with a placeholder [19]. To avoid cluttering, only the primary links are shown to users. OntoRAMA [14] enables users to browse a knowledge base (ontology) in a hyperbolic layout by duplicating cross-linked nodes. Munzner introduced a class of graphs called quasihierarchical graphs, which can be visualized using a spanning tree [30]. While the animation is striking, users may be confused because the shape of the tree changes as users interact with it. In addition, labels are hard to read because they are not aligned and often overlap. Boutin et al. used a tree-like graph as a link filtering mechanism [8]. They first extracted a spanning tree and then added some cross links to extract dense components for clustering. Force-directed algorithms were used to lay out the tree-like graph. As is often the case with other overview approaches, node labels were ignored.

Yee et al. developed an interactive exploration tool for graphs by using a radial tree layout method [40]. They animated the transition to a new layout when users select a new node. The system linearly interpolates the polar coordinates of the nodes to help users follow the transition. Since they show all the links in the graph, the view becomes cluttered for highly connected graphs. Labeling can be problematic because nodes are arranged on concentric circles. MoireGraphs [23] uses a radial layout to display a spanning tree of a graph. A Focus+context technique is used to provide an overview of graphs. It is mainly designed for graphs whose nodes are visual elements such as images. Boutin et al. also used a radial layout to visualize a hierarchical clustered graph that is the result of multilevel clustering [7]. It is a graph of clusters, where each cluster itself is hierarchically clustered. Their transformation from a graph into a spanning tree ensures that there is no overlap between clusters.

Space-filling approaches have also been used to visualize graphs as trees. Treemaps are appropriate when showing the attribute value distributions is more important than showing the graph structure [24]. Fekete et al. displayed the tree structure of a graph with a Treemap and overlayed the cross links as curved lines on top of the Treemap [15]. Treemaps have also been extended to visualize genomic data [3]. Nodes were duplicated to support gene ontologies, which are directed-acyclic graphs.

Another approach to graph visualization is to use a matrix-based representation. Abello and Korn presented matrix and color map-based techniques to visualize phone calls made between states [1]. Van Ham used multilevel call matrices in the management of large software projects [36]. He argued that matrix-based visualizations have a number of advantages over traditional node link diagrams when users are more interested in links than in nodes. Following links while reading labels, however, can be difficult with matrices.

3 Our Approach

3.1 Plant a Seed and Watch It Grow

A useful guide to designing advanced graphical user interfaces is Shneiderman's Visual Information-Seeking Mantra [34]: "Overview first, zoom and filter, then details-on-demand." An overview of the entire data collection helps users find interesting patterns, clusters, outliers, and features. However, it is notoriously difficult to generate a good overview of large graphs. Furthermore, Blythe et al. demonstrated that "there is no best layout" and that the task and graph characteristics influence which layout will do better [5].

For cases where users are more interested in the local structure of the graph, rapid browsing, and easy reading of labels, we propose an alternative guiding metaphor: "Plant a seed and watch it grow." This enables users to start with a specific node and incrementally explore the graph, avoiding complexity until it is necessary. While only partial overviews are supported, zoom and filter and details-on-demand are still useful. Furthermore, our approach can be used to complement overview-first approaches.

A similar approach was very recently used in other systems. Heer and Boyd opted for "start with what you know, then grow" and applied it to a traditional graph layout [21]. McGuffin and Balakrishnan focused on visualizing only part of a graph by using "focus" as a temporary root to visualize genealogical graphs [29].

3.2 Design Goals

There are always trade-offs when designing an interactive visualization. In this section, we describe the rationale of our design goals.

3.2.1 Take Advantage of Human Perception of Trees

Our previous work on SpaceTree and TaxonTree suggested that interaction with and interpretation of node-link tree structures pose little difficulty for novice users and therefore interactive tree visualizations can be used for a broad audience [27], [32].

3.2.2 Make as Many Nodes Readable as Possible

Many tasks involve reading the labels of nodes. For example: find and review

1. the nodes adjacent to a node,
2. the nodes accessible from a node,
3. the nodes adjacent to two nodes,
4. the shortest path between two nodes,
5. the nodes having a specific attribute value, and
6. the nodes connected only by certain types of links.

Other examples include: list all the labels in a subgraph and follow a path. For each of these tasks, users need to read labels to make sense of the data. Users will scan names in social network data to see if they know anyone, determine if there seem to be more women than men, or look for Asian-sounding names. They will also review color codings and icons associated with the nodes to judge the distributions of attributes.

3.2.3 Maximize Stability of Layout

Stability is a very important aspect of interactive layout algorithms. In fact, one of the main problems of the commonly used force-directed layouts is that they are highly unstable (i.e., the same graph might get drawn differently depending on initial conditions that are not under user control) [22]. To make the tree layout completely stable, two approaches are possible. First, we could keep the structure of the tree fixed once it was first extracted from the graph. The main drawback to this approach is that cross-linked nodes would often be very far away from each other. Second, we could force adjacent nodes to be close to each other by duplicating the cross-linked nodes (as in [14]). Although this approach works well for graphs that have an intrinsic tree structure with a modest number of cross links, it is less suitable for highly connected graphs. Furthermore, the tree will grow forever if the graph has cycles. Instead, TreePlus follows a third approach where the tree structure is modified when users make a selection by moving adjacent nodes close to the selected node. Although this approach is not completely stable, it is predictable. Changes are limited, and if users happen to traverse the same series of nodes in two different sessions, the resulting layouts will be exactly the same.

3.2.4 Offer Preview before Committing

Incremental exploration requires users to make decisions about where to go based on the information they have at any given time. To increase the "information scent" available, clicking on a node provides a preview of what nodes are connected to it. The node is not expanded until the user double clicks on it.

3.2.5 Provide Multistep Animations so Users Can Follow Changes

As users incrementally navigate a structure, it is necessary to change the layout. Animated transitions help users remain oriented [25], but they can be too complex or too fast to be accurately perceived. Previous research has shown that people conceive of temporal events as discrete sequences [41] so our approach was to decompose the layout change into meaningful steps. This was also inspired by our successful experience with SpaceTree [32].

4 The TreePlus Interface

TreePlus combines a tree-style layout, an adjacent nodes preview, and multiple custom interaction techniques to explore graphs that can be directed and contain cycles. Animation, zooming and panning, and integrated searching and browsing help users understand the graph. Users navigate the tree by double clicking on nodes in the tree browser on the left, and preview adjacent nodes on the right by single clicking on a node to bring it in focus (Fig. 1). TreePlus uses a classical tree layout by Walker [37]. The children for each node are left justified, so it is easy to scan, read, and count them. Nodes can be grouped and sorted by various ordering criteria.

We describe TreePlus using a food web data set [33]. A food web describes the feeding relationships among organisms in a community. Most animals are part of more than one food chain (or path of nodes) and eat more than one kind of food. These interconnected food chains form a complex food web. Food webs are directed, meaning that the links between nodes have specific directions (A eats B). They also can contain cycles, i.e., a node may be revisited when following a path in one direction from that node. For example, herrings eat mosquito larvae, which grow up to feed on harbor seals, which, in turn, feed on herrings. Unlike data sets used in many previous tree layout graph visualizations, food webs have no intrinsic tree structure, so they pose a greater challenge for a tree-like visualization.

4.1 Transforming Graphs into Trees

We transform graphs into trees by extracting a spanning tree. The first step is to identify a root. Domain specific default roots might exist. For example, gene ontologies have an explicit root and a canonical tree structure. Web sites have a home page. If the graph does not have an explicit root, we provide two possible defaults as suggested in [6]: 1) the node that has the most links and 2) the node whose cumulative distance to all other nodes is minimal. Users can change the root at any time; and it can be saved in the preferences. TreePlus builds a spanning tree from the root by a breadth-first search, ignoring the direction of links.

4.2 Showing Hidden Graph Structure

When we visualize graphs as trees, many cross links will inevitably become hidden, particularly in highly connected graphs. The success of a tree layout approach depends on how well the system represents those cross links. TreePlus highlights and previews adjacent nodes when a node is focused by a single click, updates the tree structure when a

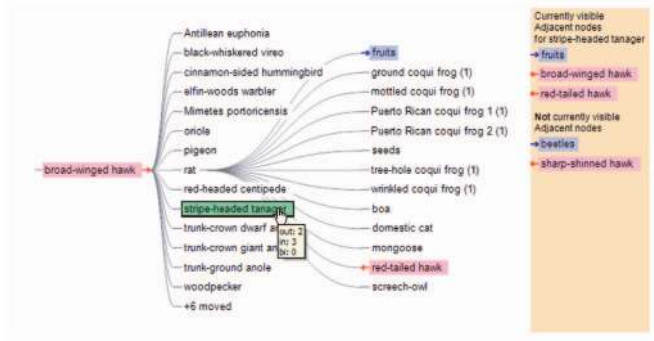

Fig. 2. "Broad-winged hawk" was set as the root, and users selected "rat" which added all its adjacent nodes to the tree. A single click on "stripe-headed tanager" gives it the focus and shows a preview of its adjacent nodes in the preview panel on the right. The adjacent nodes already present in the tree are highlighted in the tree revealing that "fruits," "red-tailed hawk," and "broad-winged hawk" are connected to both "rat" and "stripe headed tanager." Color indicates link direction.

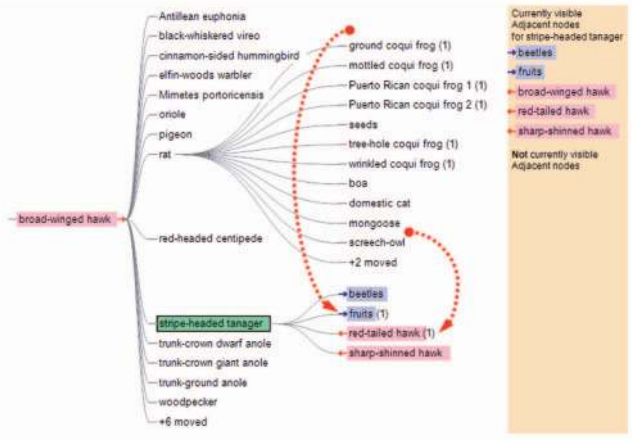

Fig. 3. Once users open "stripe-headed tanager" by double clicking, the tree is expanded to show all its adjacent nodes as its children and parent. (The dotted arrows were added to this figure to illustrate node movement.)

node is opened by a double click, carefully animates the transitions, and provides hints about the graph structure.

4.2.1 Highlighting and Preview of Adjacent Nodes

When users click on a node, the node gets the focus, indicated by a green background and thick border. In the example of Fig. 2, "stripe-headed tanager" has the focus; a list of its five adjacent nodes is shown in the preview panel on the right. Three of these nodes already appear in the current tree display, and are therefore highlighted in color on the tree. Users can see that "fruits," "red-tailed hawk," and "broad-winged hawk" are directly connected to "stripe-headed tanager" (as indicated by the highlighting) *and* to "rat" (as indicated by the tree layout). Changing the focus rapidly by using the arrow keys to go up or down a list of nodes allows users to systematically review links that are not apparent in the tree layout.

The color of the node background and arrows indicates the direction of links relative to the focus node. TreePlus uses the color blue for outgoing links, red for incoming links, and purple for bidirectional links. For example, in Fig. 2, red nodes (e.g., "broad-winged hawk") eat the "stripe-headed tanager," while the "stripe-headed tanager" eats blue nodes (e.g., "fruits").

4.2.2 Animated Update of the Tree Structure

When users double click on a node (i.e., make a new selection), the tree is expanded to include all the adjacent nodes. For example, when users open "stripe-headed tanager," two new nodes are added to the tree ("beetles" and "sharp-shinned hawk") while the nodes "fruits" and "red-tailed hawk" move from being children of "rat" (Fig. 2) to being children of "stripe-headed tanager" (Fig. 3). This change corresponds to the assumption that users are more interested in the node they last opened. Because it is a parent node, "broad-winged hawk" remains where it was in the path.

To help users maintain context, the tree is animated to its new layout (e.g., from Fig. 2 to Fig. 3) in three steps. First, TreePlus makes room for the new nodes by translating parts of the tree and creating empty space. Next, the nodes that need to move within the tree structure (i.e., "fruits" and "red-tailed hawk" in our example) move to their final position as the children of the new selection. Finally, the nodes of the preview panel move to their position in the tree. Once the animation is over, the preview panel is refreshed to reflect that all nodes are now visible. (A video at http://www.cs.umd.edu/hcil/treeplus more clearly illustrates this interaction.)

When nodes have to move within the tree structure, TreePlus leaves a trace to indicate that a move took place. The label "+2 moved" shown as the last child of "rat" in Fig. 3 indicates that two nodes have moved. Bringing the cursor over this "+2 moved" label highlights the nodes that have moved ("fruits" and "red-tailed hawk"). Users can also see that "fruits" and "red-tailed hawk" have moved once because of the "(1)" on the right side of the labels. When this number grows large it indicates that the node is linked to many of the nodes users had selected during their exploration. To be reminded of what those nodes were, users can single click on the node to bring it in focus. To close a node that is fully opened (e.g., "stripe-headed tanager"), users need to double click on it. They can use a menu item to close partially opened nodes (e.g., "broad-winged hawk" and "rat").

4.2.3 Visual Hints of the Graph Structure

During graph exploration users may want to follow a path based on the attributes of the nodes, such as the number of outgoing links. In TreePlus, users have the option to preview how fruitful it would be to go down a path. Color bar graphs placed below the nodes represent how many organisms are reachable in each direction (Fig. 4). Users can also see how many levels they can go in each direction by counting the number of white ticks. For example, if users

Fig. 4. Colored bars give a preview of how fruitful it would be to follow a path in each direction. "Broad-winged hawk" is a start of a chain since it does not have a red bar (nothing eats it). "Fruits" is an end of a chain since it does not have a blue bar (fruits eat nothing).

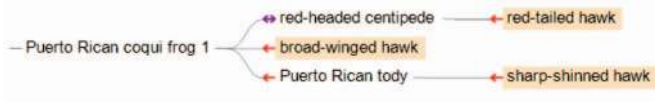

Fig. 5. A search for "hawk" with "Puerto Rican coqui frog 1" set as the root shows that the frog is eaten by the "broad-winged hawk" and indirectly by the "red-tailed hawk" and "sharp-shinned hawk."

follow the "wrinkled coqui frog" path, they will reach the end of the food chain after opening up to two levels. Similarly, they will reach the start of the chain after opening up to three levels.

4.3 Sorting

Children are depicted with a vertical list. By default this list can be sorted by name (nominal attribute), the direction of the links relative to the parent (categorical), and the number of links (quantitative). Other application-dependent sorting attributes can be added. Within each category, nodes are sorted by alphabetical order.

4.4 Search

TreePlus provides support for search. Typing a word and pressing the "Go" button displays the search results colored in beige and restricts the view to the nodes relevant to the search results (Fig. 5). In order to get a valid shortest path, a desired link direction can be specified. To find connections between two arbitrary nodes users can search for one node and set it as the root, then search for the second node.

4.5 Partial Overview

Even though TreePlus was not aimed at providing complete overviews of large graphs, it can generate partial overviews by automatically expanding the tree from any starting node, for each direction, at a selected level of expansion. For example, starting with "broad-winged hawk" and expanding as far as possible (here, level 4) with outoing links, we can reach 89 nodes and 537 links (Fig. 6). The tree overview allows users to rapidly scan labels and estimate the number of nodes and the path lengths. Users can pan to read all labels and zoom out to see everything at once. Clicking on a node and then navigating with arrow keys allows users to get a quick idea of the graph structure. For comparison, we show in Fig. 6, below the TreePlus overview, the same partial overview with a traditional graph layout (using GraphPlus, see user study section), here zoomed out to fit the narrow column width of the paper. TreePlus may use more space and not show all cross links at any given time but makes other aspects of the graph more visible.

In contrast with other graph visualizations aimed at providing complete overviews that reveal clusters and connected components, TreePlus focuses on providing local overviews. Note that if a complete overview with both directions is generated, there may not be a valid path from the root to some of the nodes. For example, in a spanning tree with "rat" as the root, the path, "rat"—"mongoose"—"plants" is not a valid food chain because "mongoose" eats "rat" and "plants." In other words, though TreePlus builds a spanning tree by a breadth-first search, some of the paths from the root to nodes are not meaningful shortest paths.

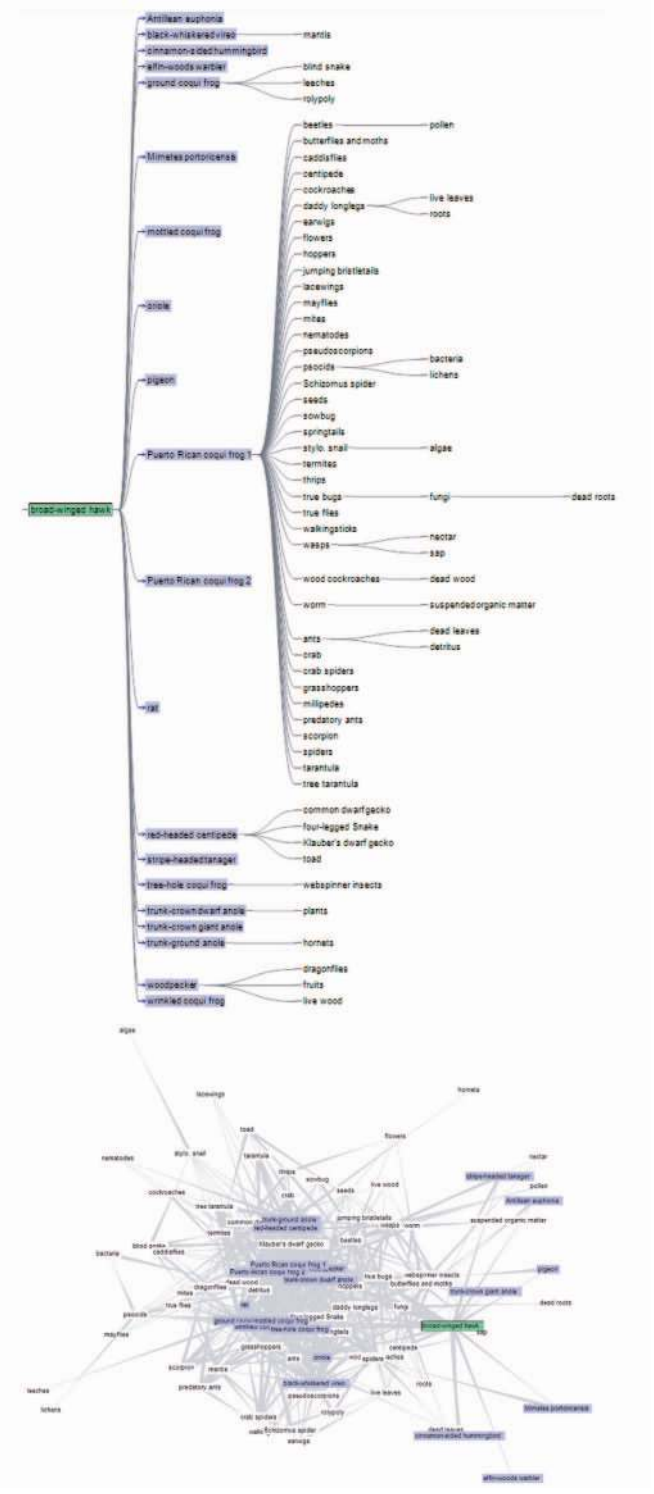

Fig. 6. Partial overviews of the graph consisting of the reachable nodes from "broad-winged hawk" with outgoing links. It contains 89 nodes and 537 links. (Top) TreePlus layout: every path from the root to nodes is a valid shortest path between the root and the node; (Bottom) The more traditional graph layout of GraphPlus for the same data. Note that the figure of GraphPlus appears smaller than TreePlus because of the different aspect ratio.

4.6 Implementation

TreePlus is a reusable widget implemented in C# with Piccolo.NET, a shared source toolkit that supports scalable structured 2D graphics [4], [31]. It is pluggable into any

.NET application. It provides APIs and fires events to communicate with the containing application.

Data structures and basic operations related to graphs are implemented in a separate library called GraphLibrary so they can be reused by other controls, such as GraphPlus. While the visual tree structure is constructed incrementally as users browse the graph, the entire graph is read into memory (in an abstract graph structure) using GraphLibrary to perform a few necessary graph operations. For example, to provide visual hints that show how many nodes can be reached in each level from a node, we computed all pairs shortest path by using the Floyd-Warshall algorithm [11]. To get the number of nodes of distance d from a node, we counted the nodes whose shortest path length from the node is d. Since the time complexity of the algorithm is $O(|n|^3)$, where $|n|$ is the number of nodes, it takes too long when $|n|$ is large. Therefore, we currently disable this feature if $|n|$ is larger than 1,000. To overcome this problem, we should precompute these numbers. For search, as described before, TreePlus shows the shortest paths from the root to the search result nodes. We first apply string-match search over the whole graph. Next, we collect all the nodes included in the shortest paths from the root to the search results. We then build a tree using these nodes and links between them, and visualize the tree in TreePlus.

Each node contains the tree structure that is used to compute a layout (parent and an array of children). In addition, each node contains two lists of cross-linked nodes (one by outgoing links and the other by incoming links). When users make a new selection, TreePlus updates the tree structure to make all the adjacent nodes of the new selection to be its children. This may result in updating the lists of cross-linked nodes.

5 Usability Study

To identify major usability issues, we conducted a preliminary usability study with two biologists and three computer science graduate students. Biologists used the sample food web whose density was about 25 percent and computer scientists used a randomly generated graph of 200 names and 3,600 links (30 percent density). As a result of this study, we made many changes to TreePlus from our earlier version [29] including:

1. Mouse click instead of mouse over now changes focus.
2. Arrows are now placed in a better location.
3. Search can now be limited to one selected direction, so as to result in a valid shortest path for that direction.
4. The preview panel was reorganized.
5. "+N moved" was added to indicate when a list of children is incomplete because some of them have moved.
6. Lines are now curved rather than straight to minimize occlusion by nodes.

6 Controlled Experiment

We conducted a controlled experiment to determine if TreePlus could outperform a classic graph visualization for certain tasks and graph densities. We used a $2 \times 2 \times 6$ (two interfaces with two densities of the graphs for six tasks) repeated-measure, within-subject design. To control for the effect of order and learning, we prepared two sets of graphs with equivalent tasks of similar difficulties, and counterbalanced the order of presentation of the interfaces and the set of graphs. We kept constant the order of densities (from low to high) and the order of tasks (from simple to complex). Four dependent variables were collected in this study: Completion Time, Success Rate, Error, and User Confidence. Success rate is the percentage of tasks correctly answered. Error was computed as the difference between the correct answer and participant response (which was possible only for two tasks requiring users to count). Participants indicated their confidence in their answer for four tasks, and completed satisfaction and preference questionnaires.

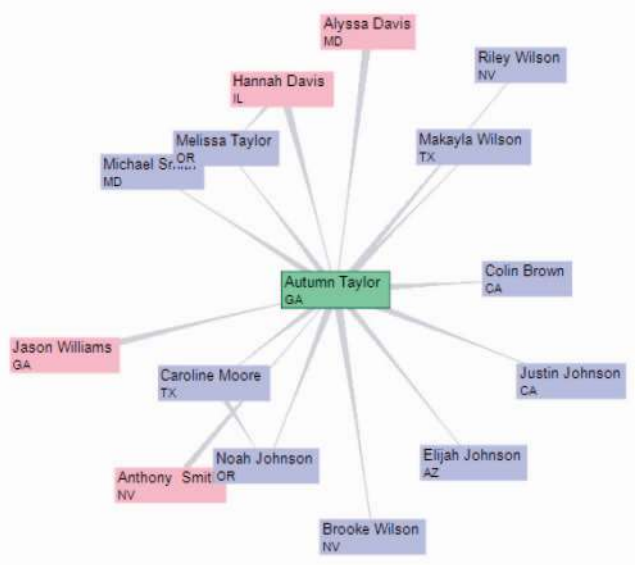

Fig. 7. GraphPlus, showing one of the displays used in a connectivity task of the experiment. "Of all the people who emailed with 'Autumn Taylor' click on the one who is email contact with the most of the others."

6.1 Interfaces: TreePlus and GraphPlus

We implemented GraphPlus (Fig. 7) using the TouchGraph [36] layout algorithm. TouchGraph is a commercial product but one of the early versions provided an open source version of the layout algorithm. This algorithm was chosen because it was designed for incremental exploration, and does a good job at avoiding occluded labels by repositioning nodes during layout. When users open a node, a new layout is recomputed to accommodate newly introduced nodes, but the TouchGraph layout algorithm tries to minimize the movements of the nodes that were already displayed. In TouchGraph, links are elongated triangles that show direction (the base of the triangle is the start and the apex is the end). However, we implemented GraphPlus to have similar features as TreePlus: It uses the same red-blue color direction coding and dynamic highlighting to reveal adjacent nodes. We also controlled the amount of time for multistep animations (1.8 seconds per complete transition). On the other hand, we considered the preview panel a major novel element of TreePlus and did not provide it in the GraphPlus interface. We wanted to compare the TreePlus interface with a state-of-the-art graph visualization interface, not solely to compare the layout algorithms.

6.2 Data and Density of the Graphs

During the usability study, we had observed that users of the food web data set would spend time reflecting whether what they saw made sense in the domain context instead of simply answering questions about connectivity. Therefore, we chose to use more neutral data sets of names for the controlled experiment.

We created two randomly generated graphs of 200 names of people. First, names were unique and taken from online lists of popular baby names in Maryland in 2004. We used only 10 popular last names resulting in 20 occurrences for each last name. To allow self cycles (links to and from the same node) and bidirectional links, and to make link direction random, we picked links among 40,000 possible ones using a pseudo-random number generator. Participants were told that each link represents an e-mail communication relationship between two people. The density of graph, d, is defined as in Ghoniem et al. [18]:

$$d - \sqrt{\frac{l}{n^2}},$$

where l is the number of links and n is the number of nodes. This definition was used because its value ranges from 0 for a graph with no links to 1 for a fully connected graph. Ghoniem et al. used graphs with three link densities (0.2, 0.4, and 0.6) with a maximum of 100 nodes. However, we wanted to use more nodes to increase the graph complexity. We chose 200 since that is the upper bound of currently studied food webs, a typical graph analysis domain. We used two link densities—low (15 percent) and high (30 percent)—that correspond to the range of densities found in real food webs. The number of links was 900 for 15 percent density and 3,600 for 30 percent. One drawback of this definition is that the number of links increases with the square of the number of nodes. So, 15 percent density might be considered very high for graphs with a large number of nodes. The 15 percent and 30 percent densities are the densities of the whole graph, and we carefully selected subgraphs that had a similar number of nodes and links.

Each node had an attribute—the US state where that person lives, randomly chosen from a set of 10 states—indicated with a two-letter abbreviation below the name.

6.3 Apparatus and Data Collection

We used a PC running Windows XP (3.0GHz Pentium 4 with 2GB RAM) equipped with an LC Technologies, Inc. eye tracking device and a 17'' LCD monitor at $1,280 \times 1,024$ resolution. Results from the eye tracking study will be reported elsewhere. The size of both TreePlus and GraphPlus was $1,280 \times 863$ (instructions filled the remaining lower screen). Since the width for the adjacent nodes preview panel was 150, the size of the main tree browser was $1,130 \times 863$. By default, labels were displayed with a 10 pt Arial font, and attributes with an 8 pt font, and users were able to zoom in and out.

Both TreePlus and GraphPlus were instrumented using the Visualization-Interaction Architecture (VIA) software [16] developed by the CogWorks Laboratory. VIA enabled the collection of all mouse clicks, mouse movements, eye movements, and systems events to a log file where they were time stamped to the nearest 16.7 ms.

6.4 Participants and Procedures

6.4.1 Participants

We recruited 28 participants (20 males and eight females) and three pilot testers (two males and one female). They were mainly computer science and engineering students who were comfortable with computers and able to quickly understand graph terminology. They already understood graph and spanning tree definitions. They received $20 for their participation. To increase motivation, a $5 bonus was given to the participant with the fastest completion time and highest success rate for each interface.

6.4.2 Procedures

Each participant used both interfaces; interface order was counterbalanced. Participants first received training on the first interface and the eye tracking system was calibrated for them. A custom-built testing program presented a series of tasks and allowed participants to complete the tasks using the first interface. Each task included two practice trials and from three to five timed trials depending on the tasks. Participants were allowed to ask questions during the practice trials, but not during the timed ones. Task descriptions were always displayed in an instruction panel at the bottom of the screen. The first two timed trials used the low density graphs and the remaining one-three trials per task used the high density graphs. Each trial had a 3-minute time limit and participants were allowed to give up a trial at any time. Once participants completed all tasks for the first interface, they answered a subjective satisfaction questionnaire. After a short break, the same procedure was repeated with the second interface. Preferences, comments, and suggestions were collected during debriefing. Each session lasted up to two hours but was typically 1.5 hours.

The tutorial was always administered by the same person following a basic script (explanations and demonstrations) and a training data set consisting of 100 names with 400 links (20 percent density). Then participants used the interface on their own and asked questions. The tutorial for the first interface—whichever it was—included some information pertinent to both interfaces (e.g., color coding, directionality, and basic interactions) so it lasted longer than that for the second interface. The training lasted about 15 minutes for the first interface, and 8 minutes for the second. In addition to demonstrating the features of the interfaces, we also included training on basic strategies to complete the tasks for both interfaces (for example, in both interfaces the strategy to find connections between two people is to search for one name, use it as the root, and search for the second name). If participants did not recall strategies, we reminded them during the practice trials.

6.5 Task Descriptions, Predictions, and Results

The six tasks and hypotheses are described below. As we predicted that each of the two interfaces would be superior for different tasks, we report independent analyses for each of the six tasks (see Fig. 8 and Table 1). The complete set of tasks is provided in the appendix available at http://www.cs.umd.edu/hcil/treeplus.

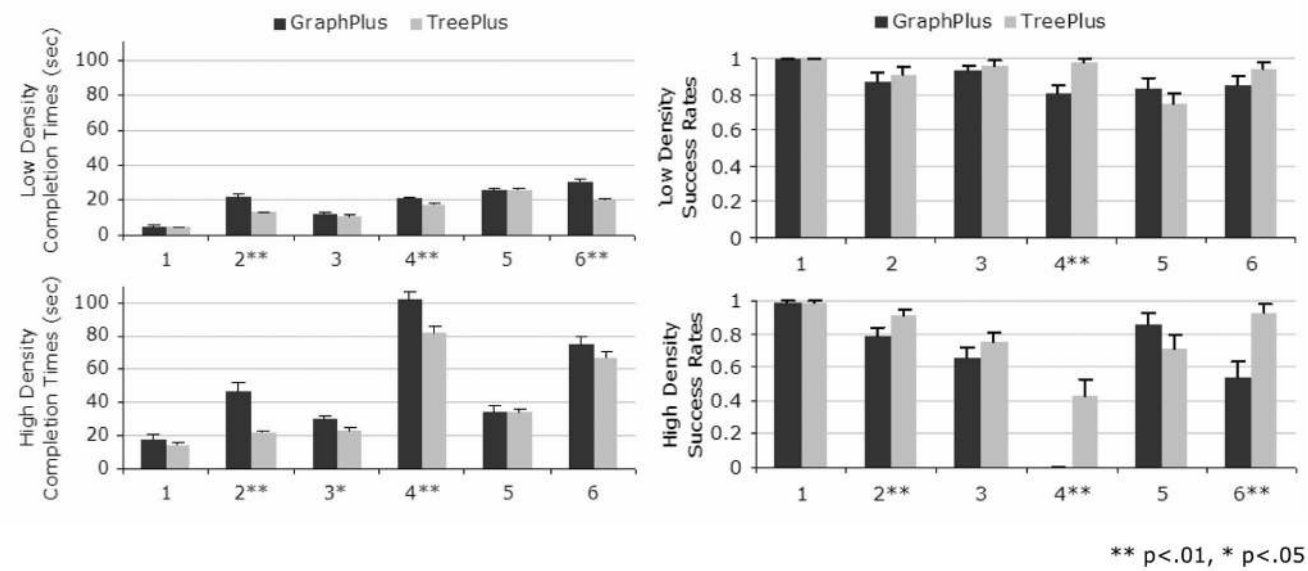

Fig. 8. Average completion times (left) and success rates (right) for tasks 1 through 6 (error bars indicate Standard Error).

6.5.1 *Find: Find a Person that Is Already Displayed: The Person Might Be Off Screen (Do Not Use Search)*

Even though this task is presented as a search task our goal was to evaluate how participants scanned the entire layout. We predicted that TreePlus would perform better because the labels are aligned and sorted, and that differences would be larger when more nodes are on the screen. However, TreePlus users might lose time panning the display, or by forgetting that nodes are grouped in categories. For two out of five timed trials, the person to be found was off screen. Completing these trials required participants to pan in or zoom out. To see whether participants benefited from the more stable layout of

TABLE 1
F-Values for Two-Factor Repeated Measures ANOVA's for Individual Tasks

		Tasks					
		1-Find	2-Browse	3-Adjacency	4-Accessibility	5-Common Connection	6-Connectivity
Completion Times							
Interface	F(1,27)	2.689	38.952**	7.497*	14.867**	0.001	8.465**
Density	F(1,27)	64.448**	72.943**	133.004**	411.862**	12.404**	167.894**
Interface*Density	F(1,27)	2.034	12.645**	2.664	9.621**	0.008	0.191
Success Rates							
Interface	F(1,27)	0	2.992-	1.871	59.565**	1.832	18.041**
Density	F(1,27)	2.077	1.991	19.017**	216.6**	0.042	6.204*
Interface*Density	F(1,27)	0	1.991	0.465	3.566-	0.186	9.084**
Error							
Interface	F(1,27)			0.004	34.784**		
Density	F(1,27)			11.238**	52.49**		
Interface*Density	F(1,27)			0.005	28.166**		
User Confidence							
Interface	F(1,27)			7.39*	20.421**	0.037	13.926**
Density	F(1,27)			27.445**	105.142**	0.028	3.174-
Interface*Density	F(1,27)			4.182-	7.652**	0.591	3.156-

** p<.01
* p<.05
- p<0.1

F-values for two-factor repeated measures ANOVA's for individual tasks (columns), with completion times, success rates, error, and user confidence as dependent variables (bold), and interface and density as the factors (rows).

TreePlus, the last trial was a repeat of a previous one ("Find again").

Contradictory to our prediction, results showed no significant differences in completion times between interfaces neither at low nor at high density. There were no differences in success rates either.

For the "Find again" trials, a two-factor ANOVA was conducted to test for Trial x Interface effects on completion times in these two trials. Although the main effect of Interface was not significant, the key interaction of Trial x Interface was marginally significant, $F(1, 27) = 3.22,\ p = 0.084$. Planned comparisons showed that completion times on the "Find again" trial significantly improved for the TreePlus interface, going down from $M = 24.54 (SE = 4.85)$ to $M = 8.57 (SE = 1.01)$, $t(27)_{\text{two-tail}} = 3.29,\ p < 0.01$, but showed no significant improvements for GraphPlus, dropping from 22.82 sec down only to 17.46 sec, $t(27)_{\text{two-tail}} = 1.06,\ p = .30$.

6.5.2 Browse: Follow a Path

To test the combination of reading and browsing we asked users to follow a path consisting of three names. We predicted that, regardless of the density of the graphs, TreePlus would work better because the nodes are easier to locate and read. The last of the four trials asked users to go back to the first node on the path. Differences between interfaces should be greatest for this last trial ("Browse and revisit") as, in TreePlus, it is easy to back-track via parents in the tree and the nodes will have moved only slightly.

The results showed a main effect of Interface and an interaction effect of Interface x Density, supporting our hypothesis that TreePlus performs better than GraphPlus and that the benefits of TreePlus increase with density. The speed advantage was not compromised by more errors as there was still a marginal significant advantage for TreePlus in success rate.

Similarly, our hypothesis that TreePlus should work better on the "Browse and revisit" trials than GraphPlus was supported as well. A two-factor repeated measures ANOVA on the effects of Revisitation (Browse versus "Browse and revisit" trials) and Interface in the Browse task revealed a significant main effect of Interface, $F(1, 55) = 37.61, p < 0.01$, a significant main effect of Revisitation, $F(1, 55) = 26.42$, $p < 0.01$, and a significant Interface x Revisitation interaction, $F(1, 55) = 14.11, p < 0.01$. Planned comparisons showed that the interaction reflected the fact that completion times for the TreePlus interface did not differ between Browse $(M = 16.39, SE = 1.17)$ and Browse+Revisit $(M = 18.36, SE = 1.10)$, $t(55)_{\text{two-tail}} = 1.45,\ p = .15$, whereas completion times for GraphPlus were significantly longer for the Browse+Revisit $(M = 43.16, SE = 4.47)$ trials than the Browse $(M = 25.98,$ $SE = 2.18)$ trials, $t(55)_{\text{two-tail}} = 4.71, p < 0.01$.

6.5.3 Adjacency: Among All Those Who Communicate with a Specific Person, Count Those with a Given Characteristic

We asked participants to count to simulate a task where all node labels have to be read. Participants had to expand the graph, scan the node labels, and be aware of the direction of the links. We marked the specific starting person with a red circle so that participants would not spend time finding it. We predicted that there would be no difference between interfaces at low density, but that at high density, GraphPlus would suffer from severe occlusion problems.

The predicted interaction of Interface x Density was not significant; however, the data showed a slight advantage for TreePlus as density increased. We might expect the advantage to be greater for higher densities than used in this study. There were no differences for success rate and error.

6.5.4 Accessibility: Count People with a Given Characteristic within Two Links (Distance 2) of a Given Person

Users had to use a menu item to expand the tree two levels down and then read and count nodes. We again marked the starting node with a red circle. We expected that the results would be similar to the Adjacency task: no difference between interfaces for the low density graph, with GraphPlus suffering from occlusion at high density. However, with TreePlus, users may also spend a lot of time panning the tree, or forget to pan the tree.

The results showed a significant effect of Interface and an interaction effect of Interface x Density, supporting our hypothesis that TreePlus performs better than GraphPlus (Fig. 8 and Table 1). The benefits of TreePlus increase dramatically with density. We also found a strong significant difference in favor of TreePlus for success rate and error.

6.5.5 Common Connection: Find All People Who Have Been in Direct E-Mail Communication with Two Given People

Nodes for the two given people were not on the screen so participants needed to use the strategy they had learned (search, reroot, and search again) with both interfaces. Regardless of link density, we predicted that there would not be any difference in time between interfaces because the search strategies are identical.

As predicted, no significant differences were found in completion times or success rates.

6.5.6 Connectivity: Find Who Has the Most E-Mail Relationships with Other People in a Group

The "group" to explore was defined as all the people who exchanged e-mail with a specific person. We again marked that specific person with a red circle. Here we expected GraphPlus to do better than TreePlus because all links are drawn on the display while TreePlus requires interaction to reveal cross links. However, for the high density graph, performance with GraphPlus may suffer due to occlusion.

To our surprise, there was a significant main effect of Interface in completion times as well as success rates that favored TreePlus (Fig. 8 and Table 1). The advantage of TreePlus increased with density.

For error, significant main effects of Interface and Density, as well as an Interface x Density interaction were found, with error being smaller for TreePlus interface, and especially small on the low-density trials.

6.6 Confidence and Preference

Except for the Find and Browse tasks, user confidence was recorded. The overall ANOVA showed significantly higher self-reports of confidence for TreePlus $(M = 8.01, SE = 0.19)$ than for GraphPlus $(M = 7.45, SE = 0.24)$, and for low-density $(M = 8.15, SE = 0.18)$ than for the high-density trials $(M = 7.30, SE = 0.23)$. Individual task comparisons revealed significant advantages for TreePlus for the Adjacency, Accessibility, and Connectivity tasks. Significant effects of Density were found for the Adjacency and Accessibility tasks, with a significant interaction of Interface x Density for the Accessibility tasks.

When asked which interface they preferred overall, 26 out of 28 participants chose TreePlus over GraphPlus. This general question was followed by a 10-item satisfaction questionnaire with ratings on a 9 point Likert scale. Individual two-tail t-tests were performed for each of the ten questions. TreePlus received significantly better ratings for Overall use, Navigation, Layout of information, Reading many labels was easy/clear, Arrows representing direction were helpful/clear, and Use of color was helpful/clear. No significant differences were found between TreePlus and GraphPlus for Predictable system response, Ease of learning, Use of highlighting was helpful/clear, and Use of arrows was helpful/clear.

6.7 Observations and Discussion

There were wide differences between participants in terms of speed and accuracy; however, despite individual variability, TreePlus performed significantly better for most of the tasks. TreePlus even outperformed GraphPlus for the Connectivity task where we had hypothesized GraphPlus would perform better. For the medium sized graphs we used, the benefits of TreePlus increased with density.

We first discuss in more detail the surprising results of task 6 and then discuss specific features of the interfaces.

Connectivity task: We were surprised to observe that in task 6 (Connectivity) many participants did not use the topology information in GraphPlus. Though all cross links are drawn and users should have been able to spot nodes with the most links—especially in the low density graphs—observations lead us to suspect that participants gave up or lost confidence using the link arrows after they used messy and poorly readable graphs in previous tasks. Instead they mostly used highlighting just as TreePlus required them to do. TreePlus even had a higher success rate, possibly because this highlighting exploration could be performed in a more orderly way. After observing this effect with many participants we considered reminding users that they could better use the links information of GraphPlus, but decided that we could not change the training procedure in the middle of the experiment.

Occlusion versus Panning: Occlusion seemed the most important problem for GraphPlus. During the most complex task (the Accessibility task for the high density graph), many participants gave unhappy exclamations such as "Oh boy" and "Wow." Furthermore, while 12 participants answered correctly with TreePlus, with GraphPlus no one answered correctly, one participant could not finish within the 3 minute time limit, and one participant gave up. For TreePlus, panning was an issue. Some participants seemed to take a few moments to remember to pan. Some tried to use the mouse wheel which was not supported at that time. Three participants made explicit negative comments about panning, and many people sighed when they had to keep panning for the very tall tree for the most complex task. Nevertheless participants were much more successful completing the task correctly with TreePlus than with GraphPlus.

To enable users to see all children of a node at once without panning, we used multicolumn layout when the height of the children of the currently opened node was bigger than the window height (Fig. 10a). However, one participant noted "Multicolumn layout was useful but confusing." Although it was not needed and did not help, many participants panned anyway in the apparent belief that there were more nodes above or below.

Search: With both interfaces, the search results showed only one shortest path between two nodes (Fig. 5). However, anecdotal evidence suggests that some participants some of the time thought that the person currently shown on the path was the only one who communicated with both people.

Preview panel in TreePlus: In contrast to our expectations, most participants did not seem to use the adjacent nodes preview for the Adjacency task. Instead, people just opened nodes. This might be because it is simple and cheap, users are not accustomed to the adjacent nodes preview, or they do not like to shift focus between two views. Among seven participants observed actively using the adjacent nodes preview, two said that they really liked it. We plan to make the preview panel optional.

Note that, for both interfaces, users could not select other nodes during the animation. However, adjacent nodes for the selected node are positioned at their new position later in the animation sequence with TreePlus than with GraphPlus. We believe that GraphPlus gained 0.6 seconds (1/3 of the total animation time) for the Browsing task because participants were able to start visually scanning earlier. In spite of this disadvantage, TreePlus still worked better than GraphPlus.

7 TreePlus Improvements since the Study

To facilitate the Connectivity task we have added an optional connectivity hint. A black vertical bar on the left side of the node shows the percentage of the connected nodes among on-screen nodes. For example, among all organisms eaten by "broad-winged hawk," "Puerto Rican coqui frog 1" has the most connections (Fig. 9). We also added the possibility to draw the cross links as lines to show the overall connectivity of the partial graph on screen. Temporarily revealing those lines might give an idea of the overall connectivity.

To address some of the problem of multicolumn layouts (Fig. 10a), we now balance the heights of columns, and guarantee a gap between the edge of the window and the multicolumn background (Fig. 10b), making it more obvious that there is no need to pan.

For search, TreePlus now shows all shortest paths between two nodes. Users can confirm the actual shortest paths by turning on the option that draws the cross links

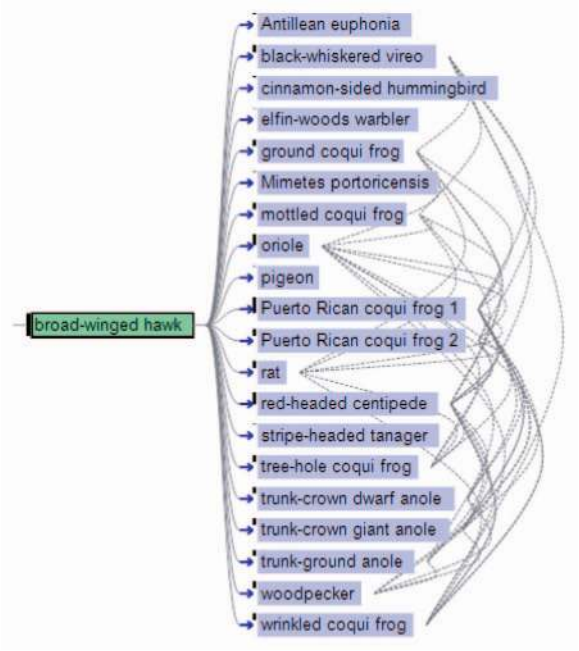

Fig. 9. Connectivity bar indicated by a black vertical bar shows the percentage of connected nodes for each node. Cross links are shown on demand with the dotted lines.

(Fig. 11). Users can also search several keywords (e.g., several peoples' names) separated by semicolon.

We also applied our techniques to other data sets, such as the coauthorship graph and citation network for the ACM CHI Proceedings [26]. These data sets introduced additional requirements. For example, each paper has a categorical attribute indicating the type of publication. TreePlus now shows the categorical attribute with a colored dot. Since paper titles are usually very long, TreePlus enables users to set the maximum width for nodes. When the label is clipped, an ellipsis indicated that there is more text which is revealed when the cursor is moved over the node.

8 Conclusion and Future Work

We have developed an interactive graph visualization system based on a guiding metaphor: "Plant a seed and watch it grow." In contrast with the more familiar overview techniques, which are effective at (but also limited to) revealing overall structure and the existence of clusters or bridges, our technique addresses the needs of users to explore parts of the graph in detail and rapidly read labels to analyze the meaning of relationships.

Our results suggest that visualization and interaction techniques can effectively support incremental exploration of a graph, and can reveal the graph structure superimposed onto a tree structure. Our user study compared TreePlus with a standard graph visualization system and found that participants completed several tasks faster and with fewer errors with TreePlus. Participants also reported higher levels of confidence in their answers with TreePlus and most of them preferred TreePlus. For the medium sized graphs used in the study, the benefits of TreePlus increased with density. Although the high density we used is representative of real food webs, it may not be as common in other domains. For

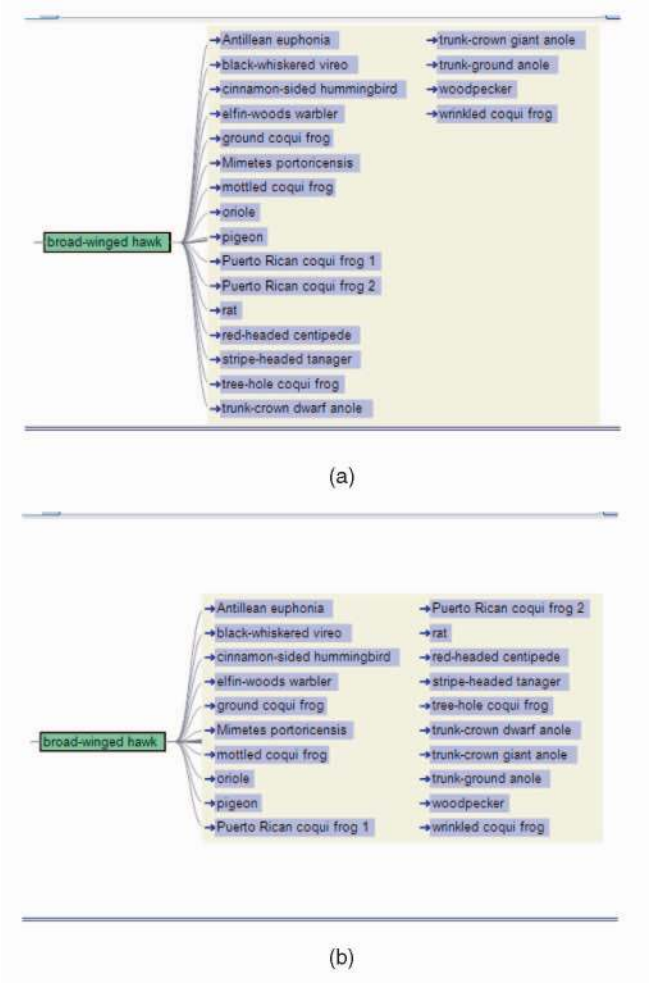

Fig. 10. (a) Previous multicolumn layout and (b) improved multicolumn layout.

large graphs, our low density (15 percent) might also be considered high and this might have contributed to the good performance of TreePlus overall in this experiment. Further evaluations using lower densities with larger graphs would shed more light on the benefits of TreePlus. The visual hints described in Section 4.2.3 were not tested in the study. It would be interesting to evaluate their benefits. A longitudinal study in a natural domain-specific context would also increase our understanding of the benefits and limitations of TreePlus.

At the start of this study, our biggest concern was that it might be confusing to look at graphs as trees. The very encouraging results of the controlled experiment lead us

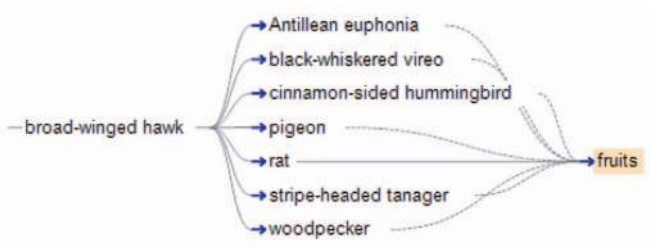

Fig. 11. A search for "fruits" from "broad-winged hawk" with cross links visible shows all shortest paths from "broad-winged hawk" to "fruits."

now to wonder if this is a natural way for people to interpret graphs. Rather than thinking about the whole graph, users might remain focused on the adjacent-node relationships, which are well represented as parent-children links in a tree. Most people who preferred TreePlus said it was not confusing to look at the tree representations. One person said that "While I was doing the tasks, I did not think of [TreePlus] as a tree." Other comments included: "I was very comfortable using it because I am used to the hierarchical structure" and "I think trees are logical and ordered arrangement of the graphs." Many participants really liked the tree layout and the alignment, grouping, and sorting of the children. Obviously further studies are needed to investigate this hypothesis which could have important and testable implications for interactive graph visualization.

ACKNOWLEDGMENTS

The authors give special thanks to Ben Shneiderman who provided encouragement and many helpful comments. They would like to thank Stephane Gamard for his help with VIA, Jean-Daniel Fekete for discussion, and Aaron Clamage for porting Piccolo from Java to .NET. This work is supported by US National Science Foundation grant #0219492 and ARDA/Booz Allen Hamilton #MDA-904-03-c-0408 Novel Intelligence from Massive Data Program.

REFERENCES

[1] J. Abello and J. Korn, "MGV: A System for Visualizing Massive Multigraphs," *IEEE Trans. Visualization and Computer Graphics,* vol. 8, no. 1, pp. 21-38, Jan.-Mar. 2002.

[2] D. Auber, "Tulip: A Huge Graph Visualization Framework," *Graph Drawing Softwares, Math. and Visualization,* Berlin: Springer-Verlag, pp. 105-126, 2003.

[3] E.H. Baehrecke, N. Dang, K. Babaria, and B. Shneiderman, "Visualization and Analysis of Microarray and Gene Ontology Data with Treemaps," *BMC Bioinformatics,* vol. 5, http://www.biomedcentral.com/1471-2105/5/84, June 2004.

[4] B.B. Bederson, J. Grosjean, and J. Meyer, "Toolkit Design for Interactive Structured Graphics," *IEEE Trans. Software Eng.,* vol. 30, no. 8, pp. 535-546, Aug. 2004.

[5] J. Blythe, C. McGrath, and D. Krackhardt, "The Effect of Graph Layout on Inference from Social Network Data," *Proc. Graph Drawing Conf.,* vol. 1027, pp. 40-51, 1995.

[6] R.A. Botafogo, E. Rivlin, and B. Shneiderman, "Structural Analysis of Hypertexts: Identifying Hierarchies and Useful Metrics," *ACM Trans. Information Systems,* vol. 10, no. 2, pp. 142-180, Apr. 1992.

[7] F. Boutin, J. Thièvre, and M. Hascoët, "Multilevel Compound Tree—Construction Visualization and Interaction," *Proc. Interact Conf.,* vol. 3583, pp. 847-860, 2005.

[8] F. Boutin, J. Thièvre, and M. Hascoët, "Focus-Based Filtering + Clustering Technique for Power Law Networks with Small World Phenomenon," *Visualization and Data Analysis, Proc. SPIE,* vol. 6060, 2006.

[9] F.-J. Brandenburg, "Nice Drawings of Graphs are Computationally Hard," *Proc. Informatics and Psychology Workshop,* pp. 1-15, 1988.

[10] I. Bruß and A. Frick, "Fast Interactive 3-D Graph Visualization," *Proc. Graph Drawing Conf.,* vol. 1027, pp. 99-110, 1995.

[11] T.H. Cormen, C.E. Leiserson, and R.L. Rivest, *Introduction to Algorithms.* MIT Press, McGraw-Hill, 1993.

[12] G. Di Battista, P. Eades, R. Tamassia, and I.G. Tollis, "Algorithms for Drawing Graphs: an Annotated Bibliography," *Computational Geometry: Theory and Applications,* vol. 4, no. 5, pp. 235-282, 1994.

[13] G. Di Battista, P. Eades, R. Tamassia, and I.G. Tollis, *Graph Drawing: Algorithms for the Visualization of Graphs.* Englewood Cliffs, N.J.: Prentice Hall, 1999.

[14] P.W. Eklund, N. Roberts, and S.P. Green, "OntoRama: Browsing an RDF Ontology Using a Hyperbolic-Like Browser," *Proc. First Int'l Symp. CyberWorlds,* pp. 405-411, 2002.

[15] J.-D. Fekete, D. Wang, N. Dang, A. Aris, and C. Plaisant, "Overlaying Graph Links on Treemaps," *Posters Compendium of InfoVis,* pp. 82-83, 2003.

[16] S. Gamard, M.J. Schoelles, C. Kofila, V.D. Veksler, and W.D. Gray, "CogWorks Visualization Architecture: Cognitively Engineering Next Generation Workstations for Decision Makers," *Proc. ACT-R Workshop,* 2005.

[17] E.R. Gansner and S.C. North, "Improved Force-Directed Layouts," *Proc. Graph Drawing Conf.,* vol. 1547, pp. 364-373, 1998.

[18] M. Ghoniem, J.-D. Fekete, and P. Castagliola, "A Comparison of the Readability of Graphs Using Node-Link and Matrix-Based Representations," *Proc. Information Visualization Conf.,* pp. 17-24, 2004.

[19] M.C. Hao, M. Hsu, U. Dayal, and A. Krug, "Web-Based Visualization of Large Hierarchical Graphs Using Invisible Links in a Hyperbolic Space," *Proc. Fifth Working Conf. Visual Database Systems,* pp. 83-94, 2000.

[20] D. Harel and Y. Koren, "Graph Drawing by High-Dimensional Embedding," *Proc. Graph Drawing Conf.,* vol. 2528, pp. 207-219, 2002.

[21] J. Heer and D. Boyd, "Vizster: Visualizing Online Social Networks," *Proc. Information Visualization Conf.,* pp. 33-40, 2005.

[22] I. Herman, G. Melançon, and M.S. Marshall, "Graph Visualization and Navigation in Information Visualization: A Survey," *IEEE Trans. Visualization and Computer Graphics,* vol. 6, no. 1, pp. 24-43, Jan. 2000.

[23] T.J. Jankun-Kelly and K.-L. Ma, "MoireGraphs: Radial Focus+-Context Visualization and Interaction for Graphs with Visual Nodes," *Proc. Information Visualization Conf.,* pp. 59-66, 2003.

[24] B. Johnson and B. Shneiderman, "Tree-Maps: A Space-Filling Approach to the Visualization of Hierarchical Information Structures," *Proc. Visualization Conf.,* pp. 275-282, 1991.

[25] C. Klein and B. Bederson, "Benefits of Animated Scrolling," *Extended Abstracts Human Factors in Computing Systems,* pp. 1965-1968, 2005.

[26] B. Lee, M. Czerwinski, G. Robertson, and B.B. Bederson, "Understand Research Trends in Conferences using PaperLens," *Extended Abstracts Human Factors in Computing Systems,* pp. 1969-1972, 2005.

[27] B. Lee, C.S. Parr, D. Campbell, and B.B. Bederson, "How Users Interact with Biodiversity Information Using TaxonTree," *Proc. Advanced Visual Interfaces,* pp. 133-140, 2004.

[28] B. Lee, C.S. Parr, C. Plaisant, and B.B. Bederson, "Visualizing Graphs as Trees: Plant a Seed and Watch It Grow," *Proc. Graph Drawing Conf.,* vol. 3843, pp. 516-518, 2005.

[29] M.J. McGuffin and R. Balakrishnan, "Interactive Visualization of Genealogical Graphs," *Proc. Information Visualization Conf.,* pp. 17-24, 2005.

[30] T. Munzner, "Drawing Large Graphs with H3Viewer and Site Manager," *Proc. Graph Drawing Conf.,* vol. 1547, pp. 384-393, 1998.

[31] Piccolo.Net, http://www.cs.umd.edu/hcil/piccolo, 2006.

[32] C. Plaisant, J. Grosjean, and B.B. Bederson, "SpaceTree: Supporting Exploration in Large Node Link Tree, Design Evolution and Empirical Evaluation," *Proc. Information Visualization Conf.,* pp. 57-64, 2002.

[33] D.P. Reagan and R.B. Waide, *The Food Web of a Tropical Rain Forest.* Univ. of Chicago Press, 1996.

[34] B. Shneiderman, "The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations," *Proc. Visual Languages Conf.,* pp. 336-343, 1996.

[35] TouchGraph, http://www.touchgraph.com, 2006.

[36] F. van Ham, "Using Multilevel Call Matrices in Large Software Projects," *Proc. Information Visualization Conf.,* pp. 227-232, 2003.

[37] Q. Walker II, "A Node-Positioning Algorithm for General Trees," *Software—Practice and Experience,* vol. 20, no. 7, pp. 685-705, 1990.

[38] C. Ware, H. Purchase, L. Colpoys, and M. McGill, "Cognitive Measurements of Graph Aesthetics," *Information Visualization,* vol. 1, no. 2, pp. 103-110, June 2002.

[39] G.J. Wills, "NicheWorks—Interactive Visualization of Very Large Graphs," *Proc. Graph Drawing Conf.,* vol. 1353, pp. 403-414, 1997.

[40] K.-P. Yee, D. Fisher, R. Dhamija, and M. Hearst, "Animated Exploration of Dynamic Graphs with Radial Layout," *Proc. Information Visualization Conf.,* pp. 43-50, 2001.

[41] J.M. Zacks and B. Tversky, "Structuring Information Interfaces for Procedural Learning," *J. Experimental Psychology: Applied,* vol. 9, no. 2, pp. 88-100, June 2003.

Bongshin Lee is a PhD candidate in the Human-Computer Interaction Laboratory at the University of Maryland, College Park. She has broad interests in human-computer interaction, information visualization, and developing user interfaces. Her research focuses primarily on designing, developing, and evaluating innovative visualization and interaction techniques. Her theses work is centered on investigating how people interact with and interpret graph visualizations.

Cynthia S. Parr is a research associate in the Human-Computer Interaction Laboratory at the Institute for Advanced Computer Studies at the University of Maryland, College Park. Formerly an ornithologist, her recent work is on food Web modeling, information visualization, and semantic Web applications for evolutionary ecology.

Catherine Plaisant received the Doctorat d'Ingenieur degree in France in 1982, and joined the Human-Computer Interaction Laboratory at the Institute for Advanced Computer Studies at the University of Maryland, College Park (HCIL) in 1987. She is an associate research scientist in the HCIL. She has written more than 80 refereed technical publications. She recently coauthored the fourth edition of *Designing the User Interface* with Ben Shneiderman.

Benjamin B. Bederson is an associate professor of computer science and director of the Human-Computer Interaction Laboratory at the Institute for Advanced Computer Studies at the University of Maryland, College Park. His work is on information visualization, interaction strategies, digital libraries, and accessibility issues such as voting system usability.

Vladislav D. Veksler received the bachelor's degree in psychology as well as another in computer science from Rutgers University, New Brunswick, in 1999. He is a PhD student in the CogWorks Laboratory in the Cognitive Science Department at the Rensselaer Polytechnic Institute. His primary research focuses on computational cognitive architectures, psychologically valid AI, learning and memory, categorization, and information foraging.

Wayne D. Gray received the PhD degree from University of California at Berkeley in 1979. He is a professor of cognitive science and director of the Cognitive Science Doctoral Program at Rensselaer Polytechnic Institute. His research is in the fields of computational cognitive modeling, human-computer interaction (HCI), cognitive task analysis, cognitive workload, and human error. He is an associate editor of the *Cognitive Science Journal* (2006-2011) as well as the *Cognitive Systems Research Journal*.

Christopher Kotfila is an undergraduate student at Rensselaer Polytechnic Institute, dual majoring in computer science and philosophy. He has been working for the CogWorks Laboratory for the last three years, studying and developing software tools for use in computational cognitive modeling.