Universally Accessible Graph Creation and Examination in Real-Time with GSK

Sean P. Mealin

August 11, 2011

Abstract

Traditionally node-link diagrams, also known as graphs, have been considered problematic for visually impaired and blind users. Our focus is to create a tool that allows a user to create and edit graphs without being able to see a screen or use a mouse. Professor Suzanne Balik and Sean Mealin collaborated on the creation of GSK, a software based tool that allows users to work on graphs in both a visual and non-visual medium. When creating the tool, we focused on designing the user interface according to universal design principles, and real-time collaboration between sighted and non-sighted users.

In this paper, we focus on the features and methodologies behind GSK. We discuss how GSK is different from other graph solutions, and the ideals behind the program design. We also focus on implementing universal design principles, and the method selected for GSK to enable real-time collaboration between users of the program.

Contents

1 Introduction

Graphs are prevalent in computer science and many other fields. They are used to model social networks, plan program dataflow, and much more. There is both free and commercial software that allows a user to easily create, edit and design graphs with the use of a mouse. Traditionally these are very visual activities that exclude visually impaired and blind users who are incapable of dragging icons around a screen. There have been attempts to enable this group of users to work with graphs; however those solutions have either had special hardware requirements, or not given full control over the graph to the user.

GSK, our software-based graph creation and editing tool gives complete control over the graph creation and editing process to sighted and blind users alike. A simple control scheme enables blind users to easily explore graphs, leveraging the same set of skills that are used when exploring a new environment in the physical world. Unique views allow sighted and blind users to precisely position nodes, enabling the creation of visually logical graphs that can be exported to an image for sharing.

The authors of the tool, Professor Suzanne Balik and Sean Mealin focused on three key ideas throughout the design process of GSK: universal design, computational and informational equivalence, and real-time collaboration between sighted and blind users. Universal design states that there should be a single presentation layer in an application that is usable by all, no matter the actual medium, such as visual or audio. Computational and informational equivalence refers to the idea that for each access method, the same information should be available, without a significant difference in the overhead needed to assimilate the data. Real-time collaboration is the ability for users to work together, without any kind of delay caused by their particular access method. GSK is unique because it achieves all three of these key ideas, and applies it to graphs, an inherently visual concept.

2 Related Work

An excellent overview of the history of accessible graph tools can be found in ‘‘Combinatory Graph Creation and Navigation for Blind People” by Suzanne Balik. [1] Arguably the most advanced tool is Deep View by Dorian Miller [2], however the user is not given control over some aspects of the graph, such as node positioning. In order to give the graph a logical layout, sighted users were required to move the nodes with a pointing device, an activity that had no analog for the non-visual user.

3 Program Design

While the program has a single unified presentation layer, we separated all functionality and elements into two abstract access methods in order to simplify user interface design. The visual user interface access method was designed to be used with a pointing device for input, and a visual display for output. The non-visual interface access method was designed to use a keyboard for input, and use synthesized speech for output. All user interface elements and functionality were required to be accessible from both access methods. For example, if a visual object could not be represented in a non-visual way, it was replaced by another object that could be accessed by all users.

3.1 Requirements

Our three key ideas, universal design, computational and informational equivalence, and real-time collaboration are all abstract concepts that can become complicated when applied to inherently visual subjects, such as graphs. To simplify the implementation for GSK, we subdivided each one into a number of requirements that the program should fulfill. After enumerating the requirements, we refined them into concrete goals that could be achieved in program code. Listed below are the requirements that we identified for each of our key ideas:

- Universal design:
  - It should be possible to activate all functionality from both a pointing device and a keyboard.
  - The interface should be able to represent all elements both visually and non-visually.
  - All commands should be simple to remember and intuitive.
  - All access methods should maintain a synchronized point of focus.
- Computational and informational equivalence:
  - Information should be conveyed to the user as simply as possible.
  - There should be no information that is specific to a particular access method.
  - Access to data should be as efficient as possible regardless of access method.
- Real-time collaboration:
  - All access methods should be active at all times.
  - Visual and non-visual feedback should be generated simultaneously.

3.2 Implementation Details

One of the first decisions that we made, was to program the tool in Oracle's Java programming language. The cross platform design of Java allows the tool to be available to as many users as possible. The next design choice that we made was to not make the program self-voicing. A self-voicing program handles text-to-speech without relying on another program, often at the cost of customizability and synthesis quality. We took the approach of allowing the user to use their preferred screen reader with the application, so that their preferred speech settings (speech rate, speech pitch, etc) remained intact. That choice also allows commercial, high quality synthesizers to be used with our program.

To facilitate communication with the user's screen reader, we leveraged the Java Accessibility API throughout our program. By setting attributes such as the Accessible Description, we were able to produce the correct audible output with any screen reader that is capable of hooking into the Java Accessibility Bridge.

When designing the core of the program, we faced several challenges. Java's SWING API has extensive support for controls that are found in traditional business applications, such as edit boxes, labels, and date pickers, all of which could not be adapted for use in a graph. We

were forced to implement our own custom objects, which would render themselves both on screen, and through verbal feedback. Due to the use of custom objects, it became necessary for us to implement a custom focus-tracking solution. We took that opportunity to build in natural command structures, such as changing focus between graph elements using the arrow keys on the keyboard, rather than using the tab key, which is typically found in traditional applications.

The use of custom objects that rendered themselves gave us the idea of creating domain-specific categories of elements that could be added to the graph. For example, under a "automata" category, there could be objects such as "start state", "state", and "accepting state". Under a "circuit" category, there could be objects such as "resistor", "capacitor", and "battery". The categories could be extended into any domain that leverages the power of graphs.

4 Semantic and Spatial Views

When communicating graphs to a user, we identified two categories of information. The first category is "semantic information", which consists of information such as which nodes are connected, directedness of connections between nodes, and type of nodes that are found in the graph. The other category that we identified is "spatial information", which consists of information such as node location relative to an arbitrary node, and patterns of node positioning relative to the graph as a whole.

In order to communicate these two categories of information to the user, we separated them into two distinct views. "Connection View" focuses on the graph semantics, and allows the user to explore the connections between the nodes. "Grid View" allows the user to focus on spatial information, and explore the layout of the nodes. Separating the information into two different views also allows the user to selectively ignore information, which may help if the user is trying to answer a specific question about the graph.

GSK is unique in the fact that spatial information is considered of equal importance as semantic information. By allowing users to set and view the position of nodes, users are able to both create graphs with a visually logical order, and obtain an overview that may enable them to identify node-positioning motifs present in the graph.

4.1 Connection View

"Connection View" allows the user to navigate between nodes with the keyboard. The user's perspective is as if they are standing on a node, and can look at the edges entering or leaving the node with the left arrow and right arrow keys. Pressing the up-arrow key is as if the user walks down the edge, and moves to a new node. Using this control scheme allows the user to rely on navigational skills developed in real life to move around the graph; the same processes that a user uses to explore a new building can be applied to exploring a new graph.

From within this view, the user can also press the enter key to set and view attributes of a graph element, such as shape, color, or size. It is also possible to add and remove connections between nodes in this view; the user is able to instantly view the changes that result from editing the graph.

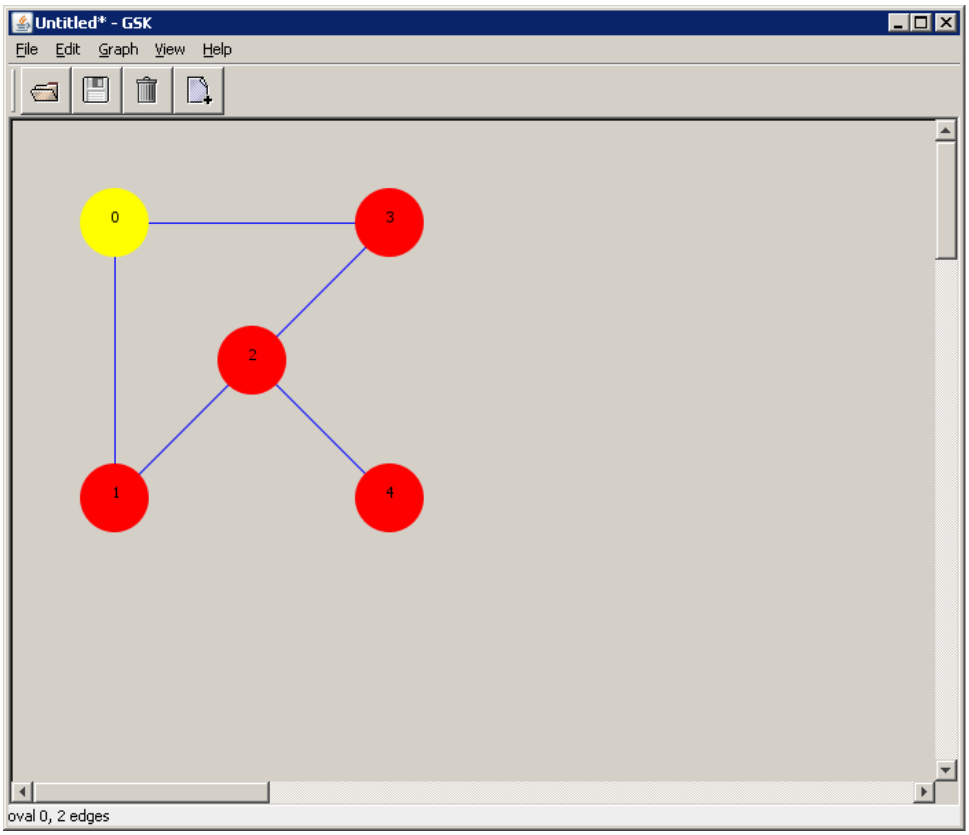

Figure 1: A simple graph in Connection View. The node highlighted in yellow is the currently focused node.

The visual interface allows users to interact with the graph in a standard way; users can use a pointing device to Drag-and-drop nodes, select a graph element to be focused, and double click to access attributes of the currently focused item.

4.2 Grid View

"Grid View" allows the user to explore how the nodes are positioned relative to one another. Just as its name implies, "Grid View" consists of a grid whose cells consist of a discrete space that can contain a node. In order to insure that the views are synchronized, each cell is automatically mapped to a predictable location in "Connection View". Changing the graph in one view will always update the other view.

The user is able to move around the grid with the arrow keys; each time a new cell is selected, feedback such as the location and whether the cell contains a node or not is relayed back to the user. When the user presses the enter key on a cell, depending on if its empty or filled, it will either add a new node, or allow the user to set and view the node's attributes.

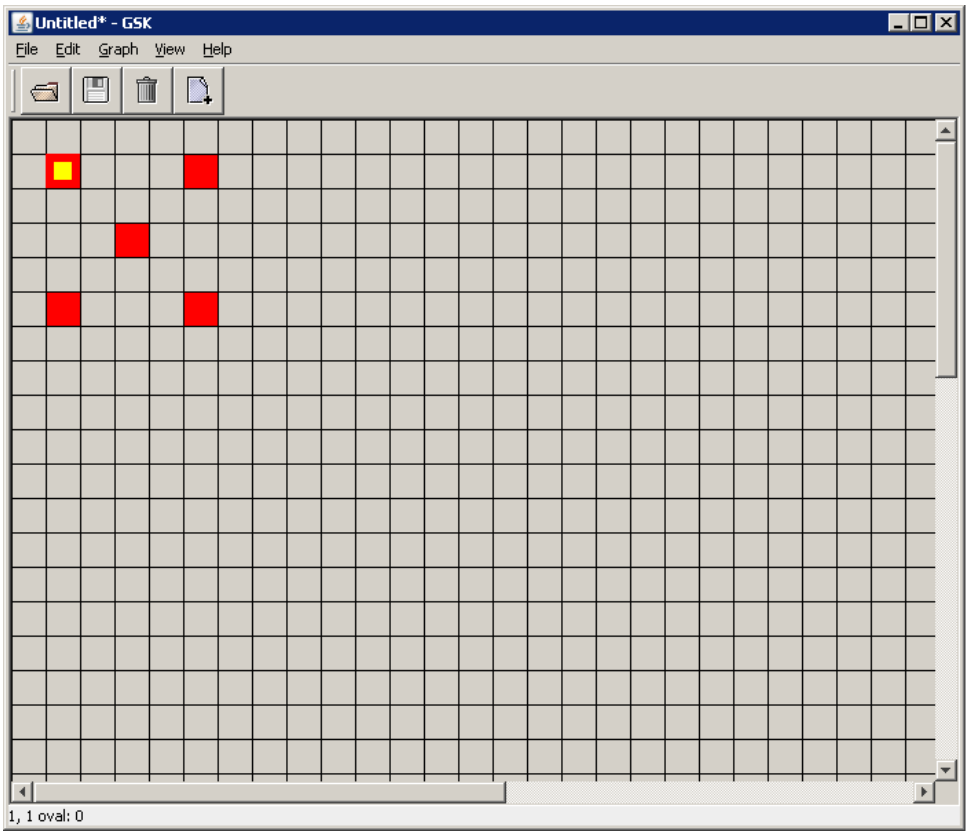

Figure 2: A simple graph in Grid View. The square highlighted in yellow is the currently focused cell.

Working with the grid also enables a user to finely position nodes to create an accurate graph.

As in "Connection View", working with a pointing device works in a standard manner; single clicking will select a cell, while double clicking will add or edit the node in the current cell.

5 Next Steps

While GSK has reached version 1.0, there are several things that we have already planned. After doing some user studies, we plan on adding some more functionality in order to make it a more useful tool.

5.1 User Studies

The next thing to do is identify any changes that need to be made in order to make the tool more useful in both an educational and professional setting. We currently plan to have a student enroll in a graph-intensive course, to track the usefulness of the tool. By getting the student to give feedback such as when the tool was useful, how well it enabled communication between the student and the professor, and additional features that would enhance the tool, we hope to be able to perfect GSK.

After making those changes, we hope to begin a large-scale user study in order to determine the effectiveness of GSK in multiple fields. We also need to get feedback on the interface, and determine if any changes need to be made.

5.2 Additional Features

As time progresses, we expect to keep on adding additional features to GSK. Some of the features that we have identified are listed below:

- The ability to add and view custom attributes on nodes and edges
- The ability to print to a Braille embosser
- The ability to create a slideshow in order to see how graphs change when an algorithm is run
- The ability to save graphs in current graph-related standard formats
- and much more...

6 Acknowledgments

I would like to thank Professor Suzanne Balik for the support and the opportunity to work on this fantastic project. I would also like to thank Dr. Richard Ladner and the Access Computing Team for funding through their generous minigrant. Lastly, I would like to thank the DREU program for the guidance they provided to make this summer as successful as it was.

References

[1] Suzanne Prem Balik. Combinatorial graph creation and navigation for blind people. Technical Report 2011-1, Department of Computer Science, NC State University, Box 8206, Raleigh, NC 27695-8206, 2011. 2

[2] Dorian Miller. *Can we work together?* PhD thesis, University of North Carolina at Chapel Hill, Chapel Hill, North Carolina, 2009.
Available at `http://search.lib.unc.edu/search?R=UNCb5970444`. 2