Roget2000: A 2D Hyperbolic Tree Visualization of Roget's Thesaurus

Jason L. Baumgartner*, Timothy A. Waugh
School of Library and Information Science, Indiana University, Bloomington, IN 47405

ABSTRACT

Thesauri, such as Roget's Thesaurus, show the semantic relationships among terms and concepts. Understanding these relationships can lead to a greater understanding of linguistic structure and could be applied to creating more efficient natural-language recognition and processing programs. A general assumption is that focus and context displays of hyperbolic trees accelerate browsing ability over conventional trees. It is believed that allowing the user to visually browse the thesaurus will be more effective than keyword searching, especially when the terms in the thesaurus are not known in advance. The visualization can also potentially provide insight into semantic structure of terms. The novelty of this visualization lay in its implementation of various direct manipulation functions, tightly coupled windows, and how data is read into visualization. The direct manipulation functions allow the user to customize the appearance of the tree, to view the density of terms associated with particular concepts, and to view the thesaurus entries associated with each term. Input data is in an XML file format. The extensibility and ability to model complex hierarchies made XML a convenient choice. The object-oriented design of the code allows for displaying virtually any hierarchical data if it is in the XML format.

Keywords: Hyperbolic Tree, Roget's Thesaurus, Direct Manipulation, XML

INTRODUCTION

Project Gutenberg in collaboration with MICRA, Inc released the 1911 version of Roget's Thesaurus as public domain electronic text (e-text).[1] Several groups have created Web based search engines to search this e-thesaurus. The ARTFL Project is a text driven search engine that allows the user to search either the full text of the thesaurus or the category name.[2] The school of informatics, City University, London, also has a text driven search engine of Roget's Thesaurus.[3] Wessler at MIT created yet another searchable version of Roget's Thesaurus. This version also includes a collapsible hyperlinked hierarchical structure of the thesaurus.[4] The search engines appear to search based on text matching. A general limiting factor of most text based information retrieval systems is proper choice of keyword search terms.[5] Improper choices will limit the success of the search. A problem with using the 1911 version of Roget's Thesaurus is that many modern technical terms are not in the thesaurus. The thesaurus also contains archaic terms that are not frequently used today. In addition, the category names were chosen by Roget and may not be obvious to a user.

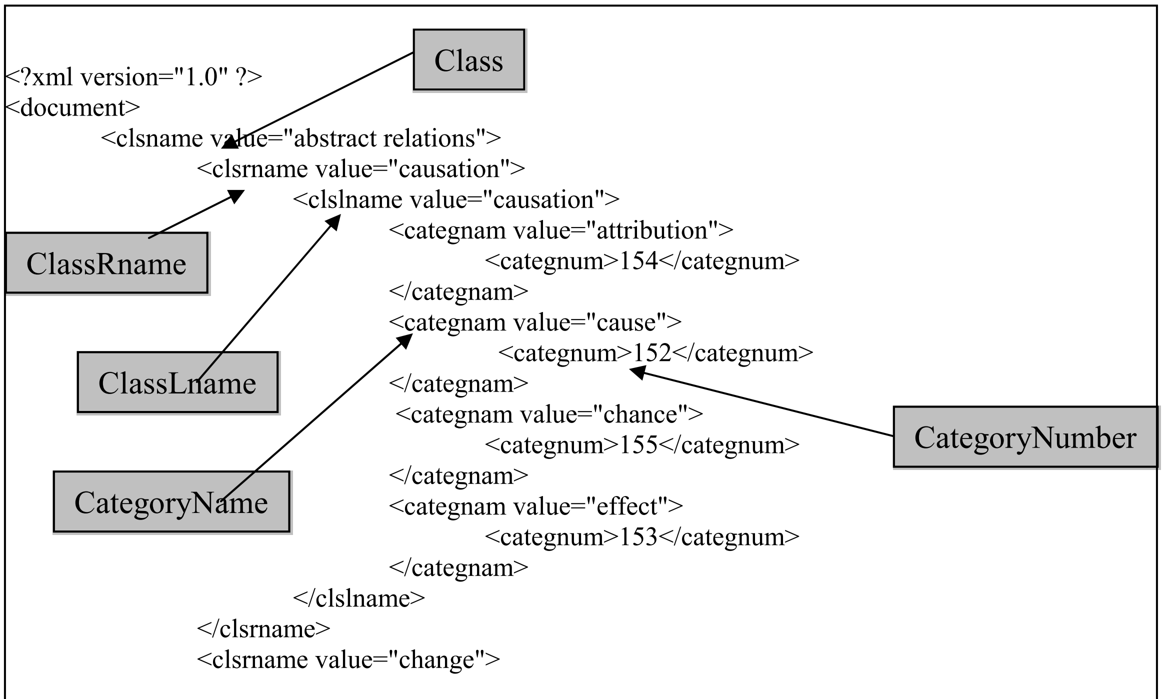
Roget2000 also uses the e-text version of the Roget's Thesaurus. It is a Web based tool for semantic research that was implemented at Indiana University by Uta Priss and John Old.[6] Roget2000 was proposed as a tool for semantic research by Dr. Rudolf Wille, founder of Formal Concept Analysis (FCA). This project follows conceptually from more than 30 years of research by Dr Walter A. Sedelow Jr. and Dr Sally Yeates Sedelow. The data used in this project is derived largely from a database version of Roget's Thesaurus built from Project Gutenberg's free-text version of Roget's Thesaurus (version 13a - 1911 - copyright information), and WordNet (developed under the direction of Prof. George Miller).

In contrast to the previously mentioned search engines, Roget2000 is designed for graphical queries and graphical output, text queries and text output, and mixed queries. Currently, only text queries and outputs are available. For the text queries a user can choose from three methods: (1) searching for synonyms of a word, (2) searching for the semantic linking term(s) between two query words, and (3) browsing the hierarchy of concepts and corresponding antonyms. Results from a synonym query are presented in a CFHTML table that shows the synonym and a ranking from 0 to 1.[7] An example of the antonym search, the antonyms *existence* and *inexistence*, are listed under the concept of *words expressing abstract relations*.[8] All queries access the thesaurus database from the Web using Cold Fusion as the middleware.

The goal of this hyperbolic tree visualization is to create the graphical query and the graphical output component of Roget2000. It is believed that allowing the user to visually browse the thesaurus will be more effective than keyword searching. The visualization can also potentially provide insight into semantic structure of terms. Since the Roget2000 hierarchy has 1000+ nodes the hyperbolic tree layout was chosen. The hyperbolic tree layout is a context + focus technique that is well suited for showing large hierarchies.[9] The primary users of the hyperbolic tree are researchers who are interested in the semantic relationships.

DATA AND PARSING

Input data is in an XML file format. The extensibility and ability to model complex hierarchies made XML a convenient choice. Upon reading the data, the Java code parses the data, creates the parent-child relationships, and creates an interactive toolbar for the direct manipulations. Although this paper deals specifically with visualizing Roget's Thesaurus, we believe that the visualization can be applied to any hierarchically organized data in XML. The use of a generalized XML hyperbolic tree implementation will be used for teaching an information visualization course at Indiana University. Figure 1 outlines a portion of the XML used by this hyperbolic tree implementation.



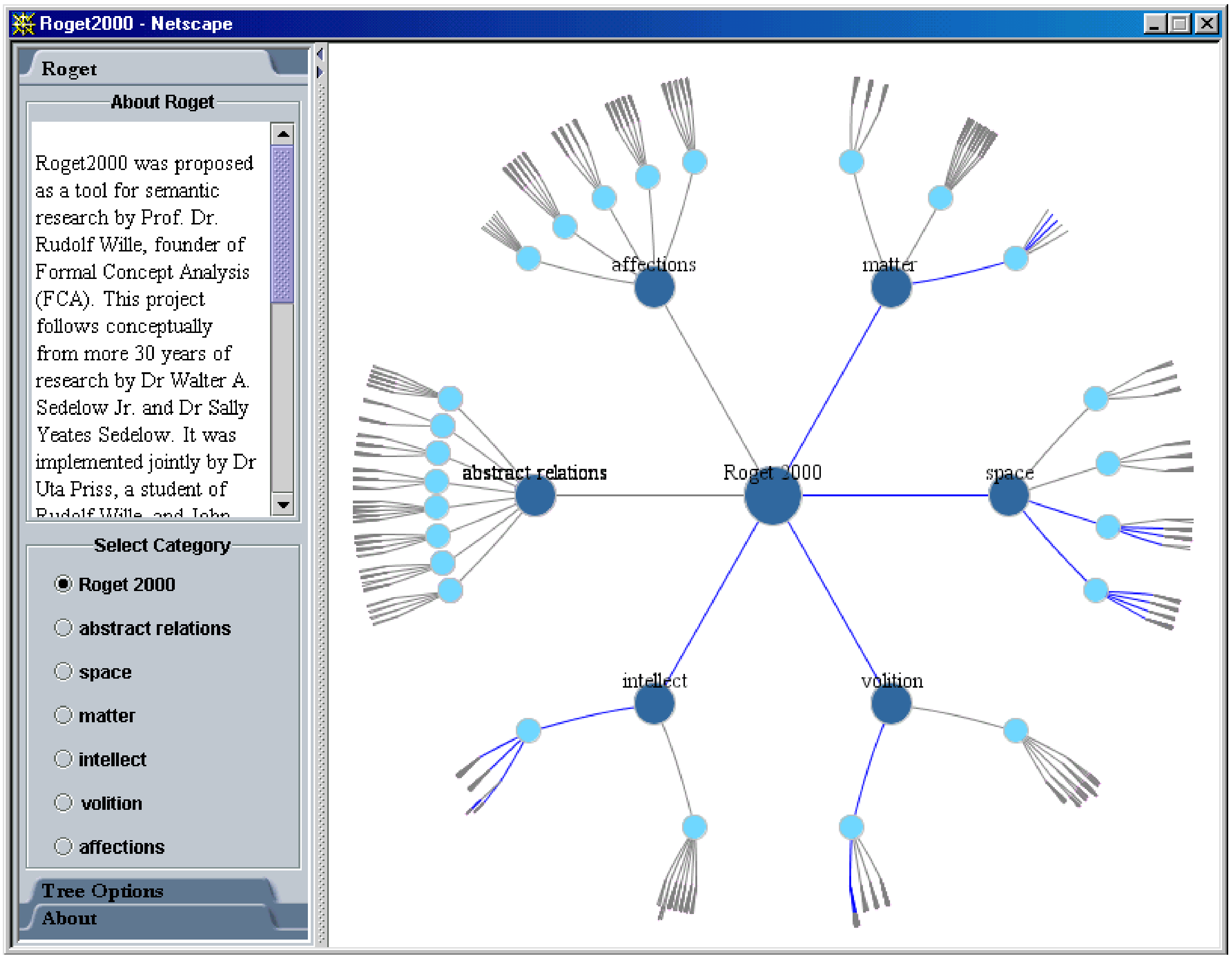
```
<?xml version="1.0" ?>
<document>
        <clsname value="abstract relations">
                <clsrname value="causation">
                        <clslname value="causation">
                                <categnam value="attribution">
                                        <categnum>154</categnum>
                                </categnam>
                                <categnam value="cause">
                                         <categnum>152</categnum>
                                </categnam>
                                 <categnam value="chance">
                                        <categnum>155</categnum>
                                </categnam>
                                <categnam value="effect">
                                        <categnum>153</categnum>
                                </categnam>
                        </clslname>
                </clsrname>
                <clsrname value="change">
```

Figure 1. Excerpt for the XML file

VISUALIZATION AND INTERACTION

The novelty of this visualization lay in its implementation of various direct manipulation functions, tightly coupled windows, etc., and how data is read into the visualization. The direct manipulation functions allow the user to customize the appearance of the tree, to view the density of terms associated with particular concepts, and to view the thesaurus entries associated with each term. The user can control various features of the tree such as the visibility of nodes, node labels, node-node connections, the shape of the nodes, and the spacing between nodes. The visualization can be accessed at http://ella.slis.indiana.edu/~jlbaumga/l697/project4/demo.html.

Due to the large number of nodes (1,000+) in the Roget's Thesaurus, a balance between rendering speed and the number of objects to be rendered was considered. Since rendering speed decreases with an increase in number of objects to be rendered, we adopted a strategy of dividing the hierarchical tree into several smaller 'sub-trees' and allowing the user to view a particular sub-tree at will. This strategy not only made the hyperbolic tree easier for a user to view information in the "focus" and the "context", but also allowed us to incorporate tightly coupled windows into the visualization. The interface has two major components, a default viewer and the hyperbolic tree display. The default viewer presents the user with a tool bar containing a list of all the sub-trees that a user can view, a tool bar for changing the appearance of the sub-tree, and a display of the tree itself. The user can view the entire Roget Thesaurus or select one of the sub-trees from the tool bar controls. The side tool bars were used because it incorporates concepts of direct manipulation into the interface. Shneiderman points out that direct manipulation gives the user a sense of control.[10]

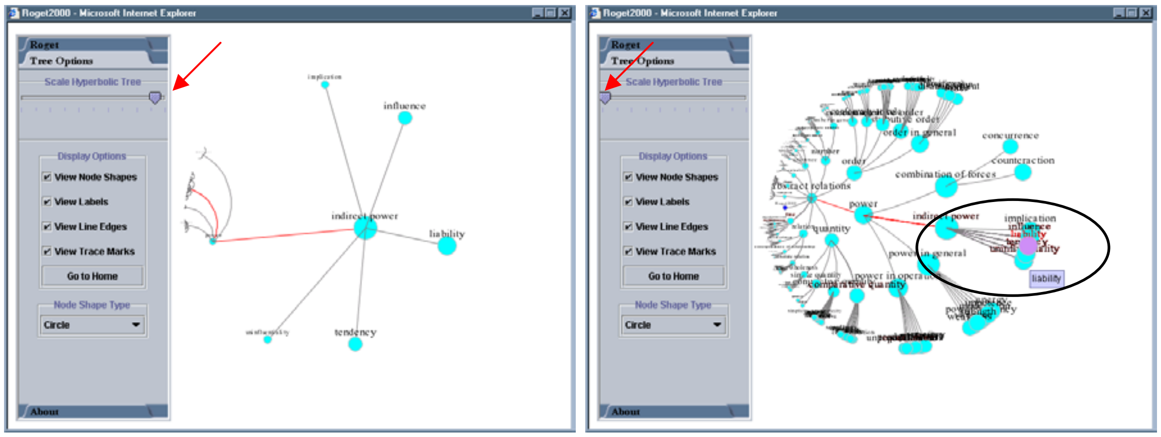


Roget2000 was proposed as a tool for semantic research by Prof. Dr. Rudolf Wille, founder of Formal Concept Analysis (FCA). This project follows conceptually from more 30 years of research by Dr Walter A. Sedelow Jr. and Dr Sally Yeates Sedelow. It was implemented jointly by Dr Uta Priss, a student of Rudolf Wille, and John

Select Category: [✓ Roget 2000] abstract relations space matter intellect volition affections

Figure 2. Initial Layout of the Roget2000 Hyperbolic Tree

Each node has its corresponding label; however, the node label is hidden at a given font size threshold. Hovering the cursor over a node results in that node changing to a selected node color and a tool tip appears that contains the node label. The presence of the blue Roget2000 root node provides a user a sense of "location" while navigating the tree. There are two methods to move nodes. One can drag a node by left mouse clicking on a node or on any white space near a node, holding down the mouse key; drag the cursor to the desired position, and then releasing the mouse key. Dragging is signified by the cursor changing from an arrow to crosshairs. One can single click on a node and that node moves to the center of the screen. Single clicking on any white space returns the sub tree's class node to the center of the screen. Figure 3 shows the result from clicking on the power node. The deferent edge coloring is part of the history function, showing the nodes that the user had been previously visited. The user has the option of turning the history function on or off.



a b

Figure 3. Moving the slider to the right (3a) increases the spacing while moving the slider to the left (3b) decreases the spacing. The circle (3b) indicates the location of the *indirect power* node.

Comparison of Figures 3a and 3b illustrates a unique feature of this visualization. Figure 3b shows that the child nodes of the indirect power node are more cluttered than the same nodes in Figure 3a. This visualization is equipped with a scaling function that adjusts the spacing between nodes. Adjusting the spacing makes the nodes and corresponding node labels more legible. The scaling function operates on the child nodes of the parent node that is in the center of the display. The scaling function operates on the child nodes (combination of forces, indirect power, power in general, power in operation) of the power node. The scale function also operates on the child nodes (implication, influence, liability, tendency, uninfluentiality) of the 'indirect power' node. The scale function is set manually by the user. The manual scale option can be found on the Tree Options tool bar. Moving the slider to the right increases the spacing between the nodes while moving the slider to the left decreases the spacing (see Figure 3a,b).

Figure 3b also demonstrates another feature of the mouse over function. The circled portion of Figure 3b is congested and difficult to read. Moving the cursor over the nodes not only changes the color of the node and label, but the node and label are repainted to the foreground making the node easier to read. In addition, the tool tip functionality is still active. Figure 3a and 3b also show that any increase in node spacing in one area of the tree is compensated by a decrease in node spacing in another area of the tree. Since the entire screen display size is fixed, there is a "conservation of display space". Any increase in node spacing must be counterbalanced by a simultaneous decrease in node spacing. The slider function uses the center node (whatever it may be) as the reference point and anything immediately surrounding it will be positively affected by the scaling and anything not immediately surrounding it will be negatively affected.

The Tree Options side bar also allows the user to further customize the tree. Checkboxes allow the user to display the tree with or without nodes, labels, line edges (node connectors), and line traces (i.e. user history). Examples of this functionality are shown in Figure 4a-f. The shape of the node can be changed to be a circle / eclipse, rectangle, or a square / rounded rectangle. The user can customize the tree by turning on or off the display labels, nodes, edges, or the trace lines (i.e. the red history lines). Figure 4d shows the tree with only the nodes appearing. This type of display is reminiscent of Holmquist et al. visualization of the World Wide Web using concentric rings.[11]

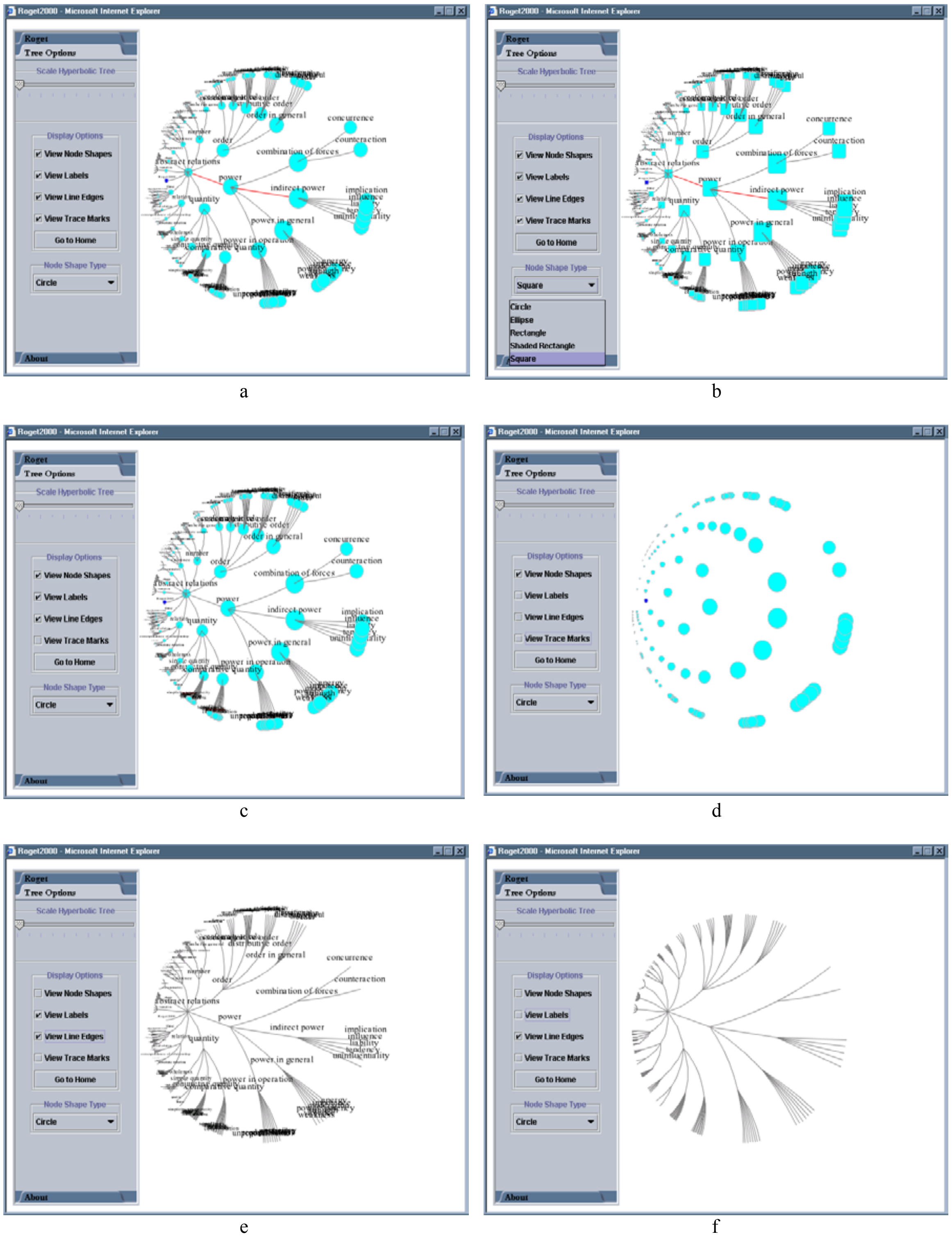
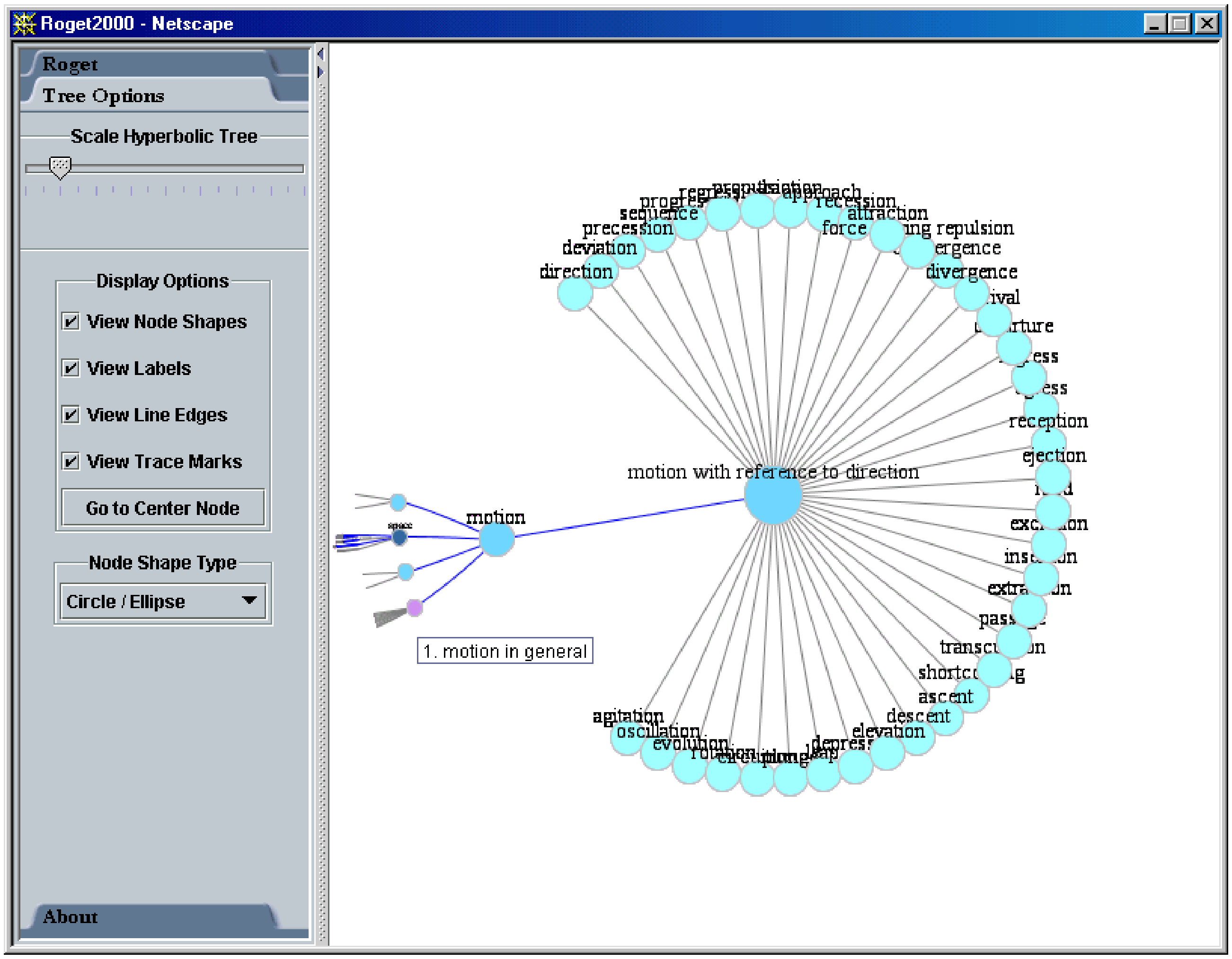


Figure 4. User defined display options. 4a shows the tree when the display node, label, edges, and trace functionalities are active. 4b shows the tree with square nodes rather than circle nodes. 4c shows the tree with the trace function turned off. 4d shows the tree when only the circle nodes are viewable. 4e shows the tree when only the label and edges are active. 4f shows the display with only the edges function active.

Selection of a leaf node results in a browser window opening that displays the corresponding Roget Thesaurus entry. The Web pages are based on the hierarchy found at www.thesaurus.com and the 1911 / 1963 version of Roget's Thesaurus.

IMPLEMENTATION

The original hyperbolic tree code of Andreas Hadjiprocopis and Vladimir Bulatov[12] formed the basis of the current hyperbolic tree visualization. While the mathematical transformations involving translation, rotation, and the placement of nodes based on an assigned angle and allocated angular width of the original code were largely kept intact; the rest of the code was rewritten. This implementation is in Java 2 and it utilized Java Swing and a model-view-controller (MVC) approach. The use of listening and command patterns allowed for the ease of direct manipulation of the hyperbolic tree and how it is rendered. For example, the shape of the node can be changed to be a circle, eclipse, rectangle, a shaded rectangle, or a square; and in future releases any shape object could be used. The user can also customize the tree by turning on or off the display labels, nodes, edges, or the trace lines (i.e. the color coded history lines). Refer to Figure 4a – 4f for examples of the direct manipulation features.



Display Options
- [✓] View Node Shapes
- [✓] View Labels
- [✓] View Line Edges
- [✓] View Trace Marks

Node Shape Type: Circle / Ellipse

Figure 5. The model is re-indexed with 'motion with reference to direction' as the root

The model for the hyperbolic tree transverses the nodes to assign an angle and an allocated angular width for each node. These values are used to calculate a complex number for the node, which represents its placement within the allocated angle that it spans. The complex number is used for placement of the node in relation to the desired distance value from its parent, and the translation and rotation functions in the hyperbolic tree. It is useful to note that the scaling function of the tree is simply a modification of the desired distance value between the parent and child nodes. Refer to Figure 3a and 3b for the tree's scaling function.

The layout function follows a pattern where a given node's angle and allocated angular width is dependent upon the number of siblings the node has and the allocated angular width of the parent. For most of the algorithm, the layout simply places each node equally across the span of the parent's angle and allocated angular width. In this algorithm when a node is selected by a click the model will re-index the nodes with selected node at the root position. All nodes' angles and allocated angular widths are recalculated with this change. This requires more time in rendering the display but it optimizes the layout of the closest nodes to the selected node.

The layout routine transverses the model of nodes to assign an angle and an allocated angular width to each node. The nodes are assigned a complex number representing their coordinates in the plane. The layout algorithm computes an angle and allocated angular width differently for each node depending on the following conditions. The model's root node is given an angle of 0.0 and an allocated angular with of $2 * \pi$. Whenever the root or selected node is a leaf node or contains only leaf children the allocation of children spans 75% of the hyperbolic tree's region while the rest of the nodes are placed in the remaining 25% of the tree's region. The distance allotted between the parent and child node will further reduce the allocation of space for the remaining tree nodes. See Figure 5 for an example of this distortion. For all other conditions the algorithm will divide a parent node's angle and allocated angular width equally amongst its children.

After the model has assigned each node with an angle, an allocated angular width, and placed it within the space then the hyperbolic tree component renders the 2D graphics to the screen. The rendering is divided among the displayable units of the arcs between nodes, the node, and the string label of the node. By rendering these elements separately the user can control which of these elements to have drawn on the screen. Refer to Figure 4a – 4f for a representation of this function.

CONCLUSION

A general assumption is that focus + context display of hyperbolic trees accelerates browsing ability over conventional trees. It is believed that allowing the user to visually browse the thesaurus will be more effective than keyword searching, especially when the terms in the thesaurus are not necessary known in advance. The visualization can also potentially provide insight into semantic structure of terms. Continued developments of this implementation have allowed for the hyperbolic mechanics to be separated from the data model and the view through the use of various interfaces. The use of the XML data structure has become generalized enough in the code so the structure could handle any XML document. This is part of an effort to build a code repository for teaching information visualization[13], and this hyperbolic tree code is a part of that effort. This code is covered under the GNU Public License for use in academic environments, with the hope that it does not infringe on the Inxight patent.

ACKNOWLEDGEMENTS

We thank John Old for supplying us with the inspiration for this project and the corresponding data. We also want to thank Katy Börner for support in developing this paper and for teaching a course in information visualization. We also thank Andreas Hadjiprocopis and Vladimir Bulatov for developing the initial hyperbolic tree Java code that formed a starting point for the development of this current visualization.

REFERENCES

[1] This quote comes from the Gutenberg Philosophy http://www.promo.net/pg/history.html (Accessed 4/2001)
[2] Gutenberg http://humanities.uchicago.edu/forms_unrest/ROGET.html (Accessed 4/2001)

[3] http://web.cs.city.ac.uk/text/roget/thesaurus.html (Accessed 4/2001)
[4] http://www.ai.mit.edu/people/wessler/thes.html (Accessed 4/2001)
[5] Sutcliffe, A. G., Ennis, M., Hu, J. (2000) Evaluating the effectiveness of visual user interfaces for information retrieval. Int. J. Human-Computer Studies 53, 741-763
[6] http://ella.lib.indiana.edu/~jold/Roget2000/index.htm (Accessed 4/2001)
[7] Go to http://ella.lib.indiana.edu/~jold/Roget2000/abc.htm and click the 'Link Queries' submit button
[8] http://ella.lib.indiana.edu/~jold/scripts/hierarch.cfm (Accessed 4/2001)
[9] Lamping, J.; Rao, R. (1994) Laying out and visualizing large trees using a hyperbolic space. UIST 94 Nov 2-4, 13-14
[10] Shneiderman, B. (1997). *Human Factors of Interactive Software*. In Designing the User Interface: Strategies for Effective Human-Computer Interaction, Addison-Wesley, 1-37
[11] Holmquist, L. E., Fargrell, H., Busso, R. *Navigating Cyberspace with CyberGeo Maps*. In Proceedings of IRIS 21, Sæby, Denmark, 1998. http://www.viktoria.informatik.gu.se/publications/98/VRR-98-10.pdf (Accessed 4/2001)
[12] http://ella.slis.indiana.edu/%7Ekaty/L697/code/hyptree.html (Accessed 4/2001)
[13] http://ella.slis.indiana.edu/~katy/L697/code/ (Accessed 12/2001)

* jlbaumga@indiana.edu; phone 812-855-0490; fax 812-855-4118; http://www.slis.indiana.edu; School of Library & Information Science, Bloomington, IN 47408