Interactive Poster:
Visualizing and Interacting with Multi-tree Hierarchical Data

Mahnas Jean Mohammadi-Aragh*
Mississippi State University

T. J. Jankun-Kelly†
Mississippi State University

Abstract

This work focuses on visualizing highly cyclic hierarchical data. A user interface is discussed and its interaction is illustrated using a recipe database example. This example showcases a database with multiple categories for each recipe (database entry).

Keywords: information visualization, focus+context, radial graph layout, database visualization

1 Introduction and Motivation

Browsing multi-tree, hierarchical data is not straightforward mainly due to its cyclic nature. Many times the user is not interested in relationships or patterns, but only in finding specific information. However, when the hierarchy is only loosely defined (or cyclic), searching is difficult without *a priori* knowledge of the underlying organizational structure. If the user already knows the exact item they are looking for, any simple search algorithm will suffice. However, when a user only has an idea or a partial query, searching cyclic relational databases becomes tricky. Most current database and hierarchical visualization methods do not readily handle loose hierarchies. This work extends current information visualization ideas to develop a visualization tool that focuses on presenting information to the user in an easily understandable fashion.

This work is related to several research areas in information visualization, including focus+context visualization, radial graph visualization, hierarchy visualization, user interfaces with drill-down, visual database exploration, and graphical glyphs. The strongest influences on this work are Zoomology [1], MoireGraphs [2], animated radial graphs [3], and Multi-tree hierarchies [4].

2 Multi-tree Hierarchies

Multi-tree hierarchies are hierarchies that contain distinguishable trees within the hierarchical structure. Multi-tree hierarchies may contain cycles. We used a recipe database to illustrate our system's functionality. The recipe database included several required data, including category, title, ingredients, and directions. Some recipes also included optional data such as how recently it was cooked (frequency), how well it was liked (ranking), and nutritional information. Most recipes have numerous categories leading to a "loosely" defined hierarchy. For example, a lasagna recipe is definitely in the "Main Dish", "Italian", and "Pasta" categories, and one could easily argue that "Main Dish" is the top of the category hierarchy, but it is not clear whether "Italian" or "Pasta" is second, since you can have Italian dishes that are not Pasta and Pasta dishes that are not Italian. The database was built with the author's personal recipes with personal rankings, recipes from books, and online recipes. The primary user task associated with this visualization is "Find a recipe".

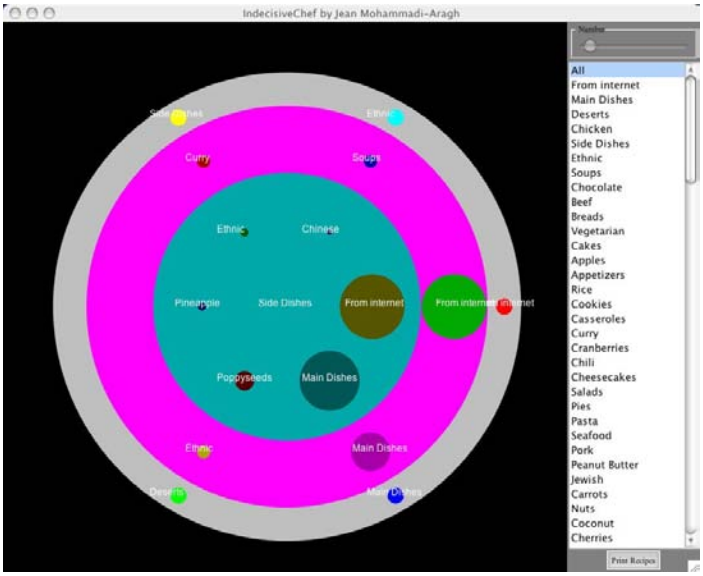

Figure 1. The system visualizing a recipe hierarchy.

3 Methodologies

Our interface (Figure 1) was developed in Java. The Java 2D API was used for the graphics and transformations (animation). The system can be divided into two sections: data layout and interaction.

3.1 Layout Method

The user interface combines a display window, where the graphics are drawn, and a tool bar, which allows for user input. The layout algorithm is a simplified version of radial graph layout; since a strict hierarchy does not exist for this database, there is no angular spread restriction. Also, there is no single root node (or focus node) rather a focus area. That means, initially, the top-level nodes are drawn in the focus area. The sub-nodes are laid out in circles, with the inner set of circles being the focus, and, thus, being larger. The space allocated to each node is 360 degrees divided by the number of nodes to-be-drawn. This step is repeated to draw the nodes from the next depth at the focus each time there is a drill-down or roll-up. Also, as the user traverses to new depths, the higher levels are pushed to the outside, away from the focus (center). Smooth animation is used to assist the user in following state changes and knowing which nodes transitioned to new locations [3]. Like MoireGraphs [2], the center of the radial layout contains the primary focus nodes; the context is laid out in

*Visualization, Analysis, and Imaging Laboratory, GeoResources Institute, Engineering Research Center, Mississippi State University, MS 39762. E-mail: jean@erc.msstate.edu.

†Department of Computer Science and Engineering, Mississippi State University, MS 39762. E-mail: tjk@cse.msstate.edu.

shrinking levels placed further and further from the center. The size differential allows the user to easily see what node is focused versus the nodes that are considered context.

Various categories are represented with various hues, i.e. the qualitative color scheme [5]. Moreover, divergent colors were chosen so the user would be less likely to misinterpret the categories as being related. Radius was used to illustrate the number of recipes "under" a specific node; hue was used in this version to differentiate between categories.

3.2 Interaction

Since all good exploratory visualization systems provide for user interaction, several interaction techniques were included. The interaction techniques involve defining a database query, undo, and obtaining details on demand. The interaction techniques are animated during level transitions (e.g. repositioning of nodes, re-querying) to aid in user comprehension.

3.2.1 Defining a query

The system starts by querying the database for all categories. The user can then select a category from the side-scrolling list to restrict the queries to a specific category. The user can add this restriction to the querying system anytime during the use of the tool. If the user chooses to not restrict the querying system, the list value "All" remains selected to show the user that there is no restriction.

The other method to develop a query is performed by clicking on the node that represents a category of interest in the display window. As with the first method, this action restricts the querying system to only return recipes where one of the recipe categories is equal to the selected node. These returned categories become new sub-nodes and are displayed in the focus area (i.e. a transition occurs). The user can continue to restrict the query by selecting sub-nodes.

To better illustrate querying, consider the recipe example. By clicking on "Chicken" in the category list, the user would restrict the system to only return recipes that were categorized as chicken. The focus nodes would consist of the top categories within the chicken category. If the user then selected the "Main Dishes" node, an AND operation would be performed on the Main Dish tree and the Chicken tree, and the new focus nodes would consist of the top categories within this new tree. This process is repeated as the user clicks on more sub-nodes. As you can imagine, this method allows the user to quickly and easily select a sub-tree of the hierarchy to explore.

3.2.2 Undo

Since a user can continually restrict the query, it is vital they be able to undo a restriction. Since the user clicks on a node to restrict a query, the opposite action, clicking on the background, un-restricts the query. Also, since the system shows context information, the user can click on an outer context ring and undo several steps at once. An undo function is required since much of exploring is trial-and-error; this interface supports quick undos and redirections.

3.2.3 Details on Demand

Once the user has restricted the query, the user needs a method to get more information on a specific data item. This is incorporated with a print button; clicking on the print button prints the current query and query results. For example, to print a recipe, the user would first restrict the query and find a recipe or a set of recipes. Then, by clicking on the print button, the recipe information (e.g. ingredients and instructions) is returned.

4 Preliminary Results

For comparison, an alternate implementation was explored. In this implementation, a strict hierarchical structure for the system was developed, and exploration was restricted to this structure. For example, top-level categories were: Main Dishes, Side Dishes, Desserts, Drinks, and Appetizers. Under "Main Dishes", the next level included Meat, Casseroles, Pasta, Slow Cooker, Sandwiches, Soups, Vegetarian, Grilling, and Ethnic.

Although a formal user study has not been completed, preliminary user feedback was collected from three distinct users regarding their preferences for the two different methods. All of the users preferred the first version of software (explained in Section 3). That version does not pre-define categories or category order of appearance. This was preferable when the user simply wanted to browse the recipes. For example, if a user wanted to look at all the beef recipes, the second implementation did not accommodate; the user was always first presented with "Main Dishes", "Sides", "Desserts", "Appetizers", and "Beverages". By incorporating a category chooser in the first version, a user was easily able to select a top category to browse or remove (by "NOT"ing the selection). Also, and probably most importantly, the first version allows the database to define the visualization rather than a programmer's pre-defined hierarchy. For another application where the hierarchy is better defined, the second version might be more applicable.

5 Discussion and Conclusions

Based on preliminary user feedback, our system improved on current methods for recipe browsing. Specifically, in our recipe database example, information was presented to the user in an easier to understand format. Without personal knowledge of the database, users were able to easily explore the database and find recipes that met their needs. Our system is appropriate for showing cyclic hierarchies. However, one limitation is that like all circular layouts on a rectangular screen, there is wasted screen space in the corners and the application does not scale well.

There are several simple extensions that could be made to our tool, such as incorporating metadata (e.g. pictures) or including "slow-in, slow-out" animation [3]. Also, alternate layout schemes could be explored. However, any new designs should ensure the user is not limited to a programmer-defined hierarchy.

References

[1] HONG, J. Y., D'ANDRIES, J., RICHMAN, M., WESTFALL, M. 2003. Zoomology: comparing two large hierarchical trees. *2003 IEEE Symposium on Information Visualization Contest Short Paper*. unpublished.

[2] JANKUN-KELLY, T. J., AND MA, K.-L. 2003. MoireGraphs: Radial focus+context visualization and interaction for graphs with visual nodes. In *Proceedings of IEEE Symposium on Information Visualization 2003*, 59-66.

[3] YEE, K.-P., FISHER, D., DHAMIJA, R., AND HEARST, M. 2001. Animated exploration of dynamic graphs with radial layout. In *Proceedings of IEEE Symposium on Information Visualization 2001*, 43-50.

[4] FURNAS, G. W., ZACKS, J. 1994. Multitrees: Enriching and reusing hierarchical structure. In *Proceedings of the Human Factors in Computing Systems CHI '94*, 330-336.

[5] BREWER, C. A. 1999. Color use guidelines for data representation. In *Proceedings of the Section on Statistical Graphics*, American Statistical Association, 55-60.