

Visualizing Dynamic Hierarchies in Tree Maps

Evren Sirin Fusun Yaman

Department of Computer Science
 University of Maryland
 College Park, Maryland 20742
 {evren, fusun}@cs.umd.edu

Introduction

There has been a great amount of research on visualizing hierarchical data like directory structures, family trees etc. Several different tree visualization techniques have been presented in the literature. Among these, TreeMap [JS91] is one of the most promising and effective visualizations. However, not every dataset has a hierarchy in its nature. In such a case user can define imaginary hierarchies among some of the attributes and feed preprocessed data into TreeMap.

One may ask the question of “Is there a benefit in visualizing a non-hierarchical data in a tree structure?” Grouping the data according to some of its common attributes may reveal patterns that may not be obvious at the beginning and may lead to additional results. There are some associated problems though; deciding which hierarchy is meaningful, how to create the hierarchies without too much work, and how to add quantitative attributes into hierarchies etc.

In this study, we tried to address those problems. We added additional features into TreeMap3 to solve the problems in visualizing non-hierarchical data. Rest of this document is organized as follows; the following two sections present motivation and related work. Section 2 describes the features we integrated into TreeMap3 on an example visualization that uses those features. Section 3 mentions future work and finally section 4 concludes the ideas presented in this paper.

Motivation

Let’s say we have a dataset that contains election results in 50 states for last three elections. The dataset also contains income and college graduation percentage, voting percentage for each state in each election year. Interesting queries are;

- In which states does the winner change?
- What is the relation between college education percentage, income and vote percentage of democrats over the years?

- What is the relation between voting percentage, college education, income and vote percentage of democrats?
- Overall what is the average vote percentage for democrats in each election?
- In each region what is the average vote percentage for democrats in each election?

In order to answer those questions we can group the data according to some attribute, then search for the answers in the groups formed.

We can create those groups by defining a hierarchy that contains the grouping information. Treemap3 is a powerful tool for displaying predefined hierarchies. One can compare the subgroups in the hierarchy by coloring or proportioning the size. Figure 1 is an example of a visualization in Treemap3 to answer the first question.

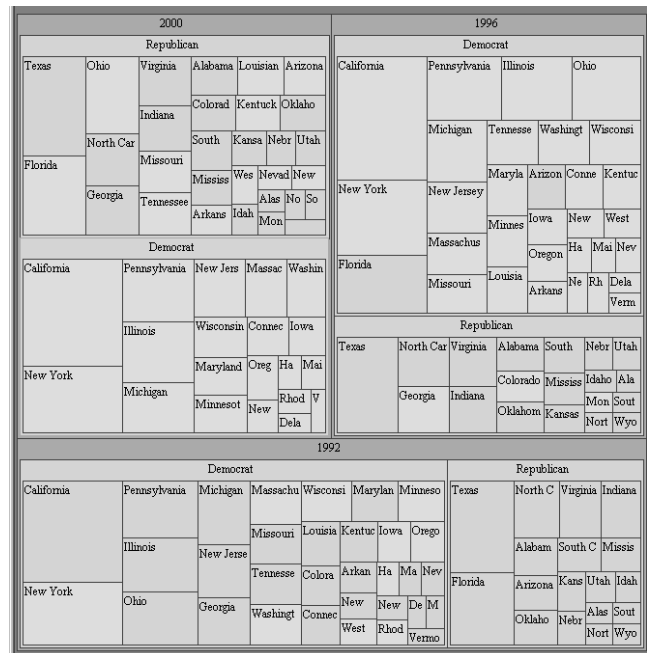


Figure 1 Visualization in Treemap to see the winner

The hierarchy in this figure contains the year and the winner. The winner of the last election is represented by colors. Blue is the republicans and pink is the democrats. The size of nodes is proportional to the electoral vote. It can be easily seen that states who voted for republicans in 1996 elections, voted for republicans in 2000 selections. Virginia is one of the states in which winner changed since 1996. Besides the information we are looking for the visualization in Figure 1 also shows how close the election results were. By comparing the total size occupied by parties in each election, one can conclude that 2000 elections was close. However the following questions can not be answered by Figure 1;

- What is the total electoral vote of parties in each election?
- What is the average income of the democrat states?

Answers to those questions and the last two questions presented at the beginning of this section require statistical information about the groups in the hierarchy. An important decision for this visualization is how to group data so that we can benefit from it most. For example in this case grouping in first by winner then by year would result the visualization in Figure 2 which would be less useful. Therefore an easy way to order and create hierarchies is necessary.



Figure 2 Another hierarchy

Another issue here is to help the user remember the path he/she has been working on previously. When there is a large number of attributes, there can be too many combinations or even permutations of them, each representing a different visualization. A useful feature is a list of past

visualizations. Maintaining a history of previously created visualizations is important since a comparison between the current visualization and previously obtained ones can be crucial [Twe97].

Now let's look at the visualization that addresses the second question (What is the relation between college education percentage, income and voter percentage of democrat over the years?). Figure 3 is a visualization in which the hierarchy contains year and income. Color is by democratic vote percentage and sizes are proportional to college education percentage. First thing to notice is, there are a lot of details and it is not easy to capture the relationship at once. This is because for every distinct value of income, a category has been formed. It would be wiser if we could define our own categories like low, medium and high income. Then we would get Figure 4 which is neat and certainly much more easy to analyze.

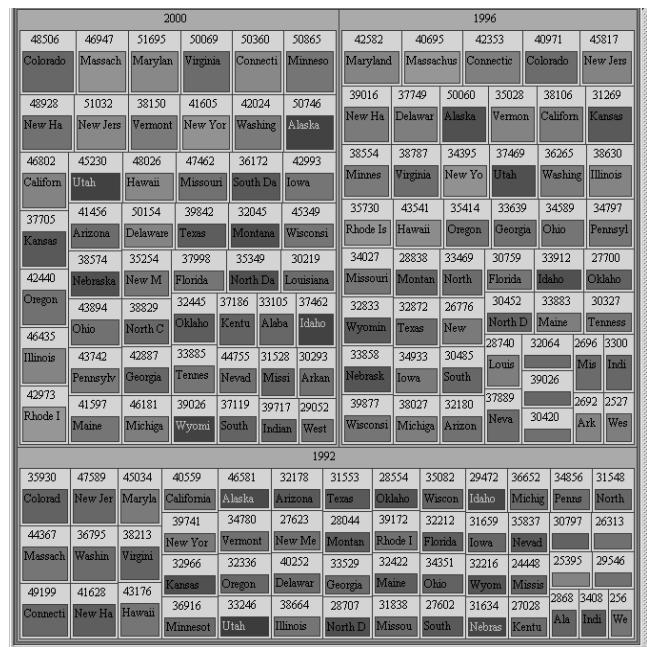


Figure 3 Visualization for question 2

When working with quantitative attributes, a major problem is how to categorize those values so that the tree will not be unnecessarily crowded. Like in this case grouping the income values into 3 bins. Therefore, a mechanism that allows and guides the user to bin those larger range or high precision values is essential. This kind of preprocessing can be performed by the user prior to loading the data into visualization system. This is how the visualization in Figure 4 is achieved. However, this method requires preprocessing for each possible bin size and number of bins. Furthermore, this approach cannot use the benefits of dynamic queries that enable the user to control the visualization incrementally.

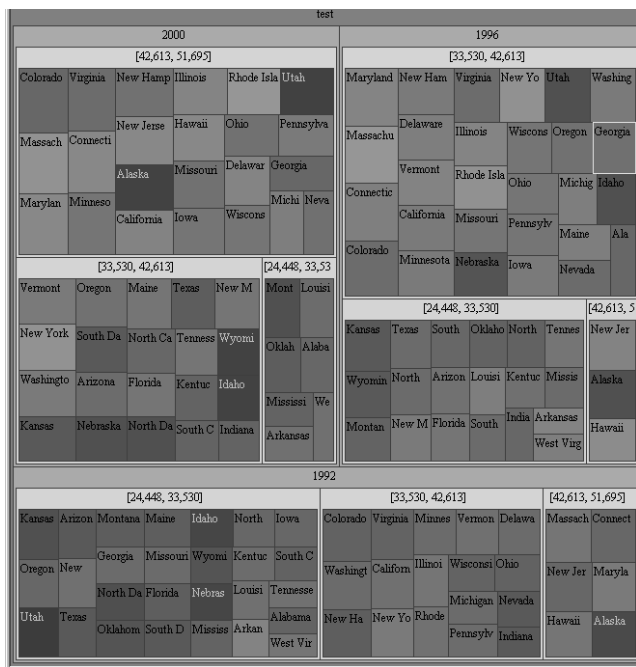


Figure 4 Another visualization for question 2

Finally, visualizing large hierarchies (like the one question 3 requires) have a different problem of their own. When the number of attributes in a hierarchy is large, it becomes harder for the user to extract valuable information from the visualization. One solution to this problem is not creating large hierarchies. This would work when the hierarchy is not fixed. If the hierarchy is fixed, then another solution may be limiting the depth of tree that is visible to the user. This comes with the complementary zooming ability. As the user zooms in, additional levels of hierarchy become visible. When the lower level nodes are not visible, it is possible to represent them in the upper levels by using aggregation methods. For example a hierarchy consists of the region and states. Let's say for each state percentage of democrat votes is displayed. Further, in the visualization if for some reason, the states became invisible then there should be a way to represent this information in the upper layer. Let's say by displaying average for percentage of democrat votes for each region.

We decided to implement our idea in TreeMap3. TreeMap displays the hierarchies using the enclosure and space filling techniques. It has powerful features to support dynamic queries, like filtering, zooming, coloring and sizing by an attribute.

Related Work

There have been extensive amount of research on visualizing hierarchical information.

- Treemaps [JS91], uses enclosure and space filling techniques to visualize the tree structure effectively.
- Hyperbolic trees [LR95], uses focus+context (fisheye) scheme for visualizing and manipulating large hierarchies. The approach is to layout the hierarchy uniformly on the hyperbolic plane and map this plane onto a circular display region.
- Cone Trees [RMC91], a 3D representation of tree structures that allows animation and 2D projection of subtrees.

TreeMaps was developed by Ben Shneiderman at the Human-Computer Interaction Laboratory (HCIL) of the University of Maryland during the 1990s. Treemaps uses enclosure to represent the hierarchy notion. Space filling technique requires minimum amount of space [Shn00]. It supports comparison of the nodes size or other attributes in the nodes by coloring or sizing the nodes proportional to the selected attribute. Filtering and reversible selections allow dynamic queries [Shn94], which lead to easier data exploration.

For visualizing nonhierarchical data, Cat Trees [KW01] by Erica Kolatch and Beth Weinstein in May 2001. Cat Trees that facilitate the dynamic visualization of categorical data. The system is built on top of an earlier version of TreeMaps. Cat Trees aims to visualize categorical data by creating imaginary hierarchies. From this point of view Cat Trees is the closest study to our work. However, Cat Trees works on data that has nominal attributes only. Even if the original data contains non-nominal attributes, these have to be categorized (binned) prior to loading the data into system. Notice that the binning performed during the preprocessing cannot be changed in the system and user has to create a new data file for every different binning he/she wants to try.

Design and Implementation

We have designed a user interface for creating hierarchies. Figure 5 displays the design. User can select from a list of available attributes, then using the "Add" button the attribute can be added into the hierarchy list. For each attribute the number of bins or distinct values and the kind of binning applied to its values is displayed in the attribute and hierarchy lists. It is possible to remove an attribute from the hierarchy by using "Remove" button after selecting that attribute. User can create permutations of the attributes in the hierarchy by moving up or down the selected attribute. Another way of creating permutations is

using the tour of hierarchies. When “Tour” button is clicked a permutation of attributes in the hierarchies is generated and for each permutation the resulting trees is displayed for 3 seconds. This feature enables the user to decide the right permutation of attributes in the hierarchy. Finally “Save” button inserts the current visualization into History. User can revisit the visualizations in history by selecting them from the dropdown list.

For quantitative and ordinal attributes user can define binning. This feature enables the grouping of similar values and reducing the number of distinct values to number of bins. In the attribute list and hierarchy list. Number of bins and the binning type is also displayed. When an attribute is selected a histogram showing the distribution of values is displayed at the bottom. If there is a binning applied for this attribute the blue separator lines are also displayed. The coordinate of these separators are written on the histogram axis. The minimum and maximum value on the axis is determined by the minimum and maximum values in the data. By clicking on the binning type in the attribute listing, user can apply a different binning. There are two predefined binning types; Equally Spaced and Equally Dense. Equally spaced binning divides the axis into equal length bins. The number of bins can be edited by double clicking in the attribute. Number of elements in a bin is displayed on the

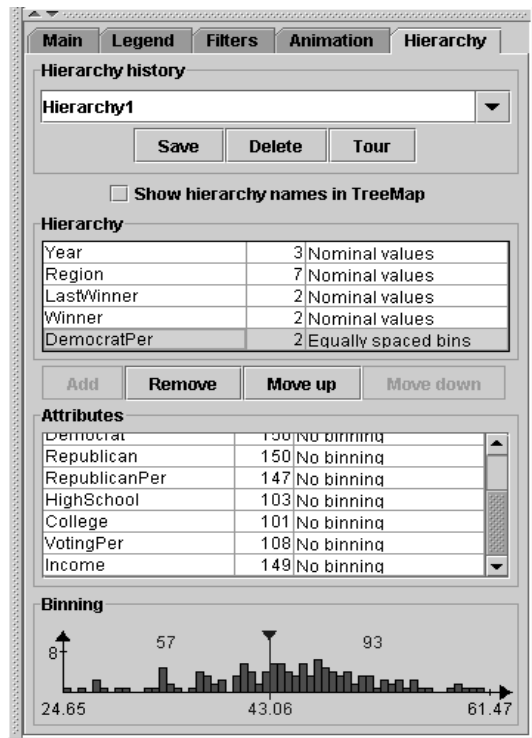


Figure 5 Dynamic Hierarchy Editor

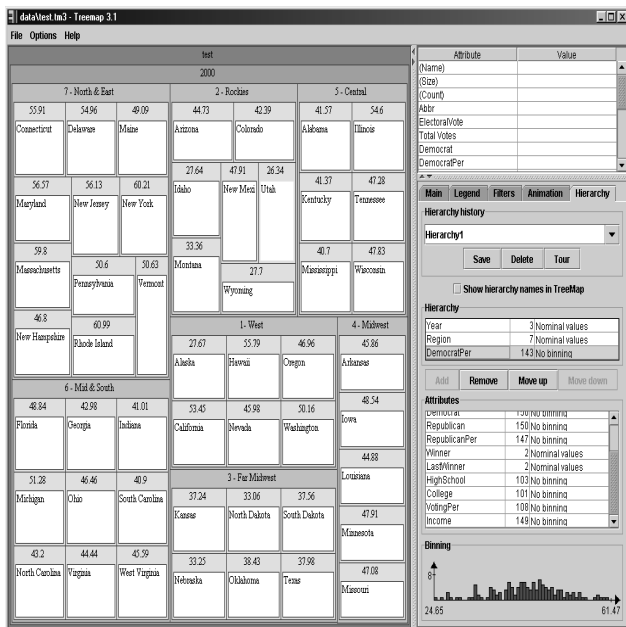


Figure 6a Treemap without binning Democratic vote percentage



Figure 6b Treemap after binning Democratic VOTE percentage

topofhistogramsforeachbin. Equallydensebinning createsbins that contain approximately the same number of bins. It is possible to create User Defined bins. User can add separators by clicking on the histogram and position these separators. An example of Equally Spaced bins is used for Democratic Vote percentage in Figure 5. There are two bins one from 24.65 to 43.06 and the other from 43.06 to 61.47. first bin contains 57 elements and the second one contains 93 elements.

Figures 6a and 6b display the effect of applying binning to quantitative attribute which is in the hierarchy. In Figure 6a democratic vote percentage is used in the hierarchy as it is, resulting in a node for each value. In Figure 6b, 3 user defined bins are created for democratic vote percentage and as a result there are significantly less number of nodes in latter tree.

When the hierarchy is large, meaning there are large number of attributes, it becomes harder for the user to understand the visualization. To reduce the detail in the visualization user may choose to filter out some data or even some of the hierarchies. Treemap3 has filtering tools for quantitative attributes. We added filtering for nominal valued attributes also (see Figure 7).

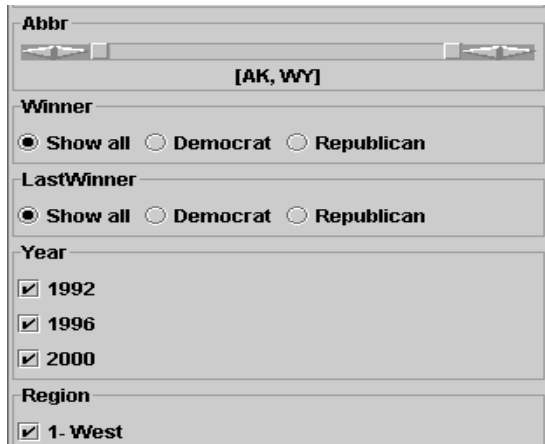


Figure 7 Filtering tools for nominal attributes

Upon seeing the large hierarchy, the user may want to limit the visible depth of the tree. In this case the tree nodes up to that visible depth limit will be drawn. Figure 8 displays a tree with depth four. The values in each node is the Democrat Percentage for each state. In Figure 10a the Maximum Visible Depth is set to 2 and lower level nodes are omitted from the visualization. However it can be seen that intermediate nodes have values displayed inside. This value is the average of Democrat Percentage in the subtree

can

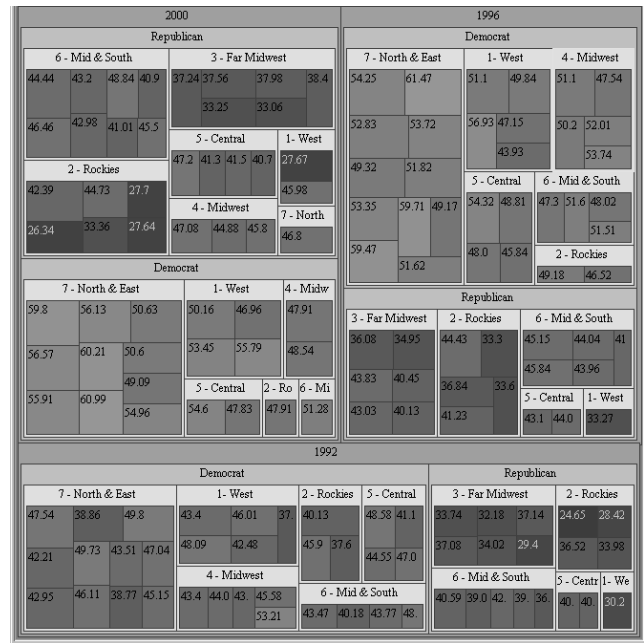


Figure 8 Whole hierarchy with non-aggregated values

represented by each intermediate node. User can select a different aggregation function for each numeric valued attribute (Figure 9). Aggregation functions we support are sum, average, minimum and maximum. To zoom in lower levels of the hierarchy the user may click on a node and additional levels of hierarchy become visible. The zoom in and zoom out functions are also available. Figure 10b displays the result of zooming in a node in Figure 10a.

Attribute	Value
(Name)	7 - North & East
(Size)	299.70000076293945
(Count)	11
Abbr	
Sum ElectoralVote	119
Sum Total Votes	21,852,973
Sum Democrat	11,968,336
Avg. DemocratPer	54.25
Sum Republican	7,387,492
Min RepublicanPer	26.82
Winner	
LastWinner	
Max HighSchool	86.9
College	0
VotingPer	0
Income	0
Year	

Figure 9 Aggregated values for a node

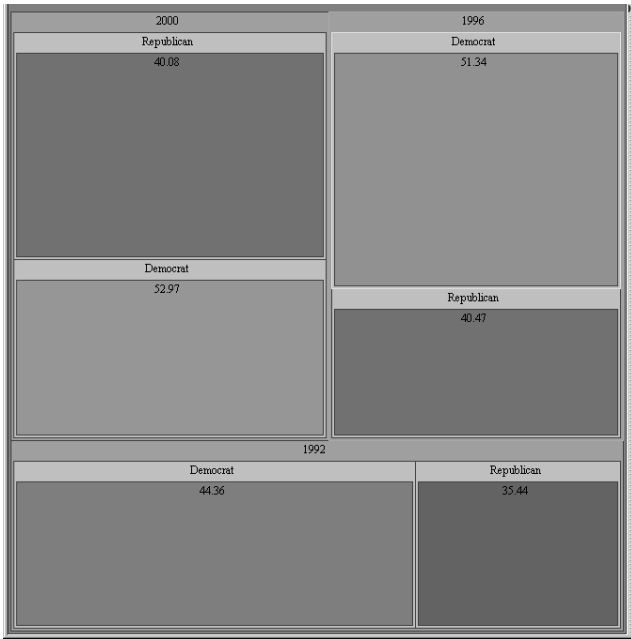


Figure 10a Twolevelhierarchywithaveragedemocrat vote

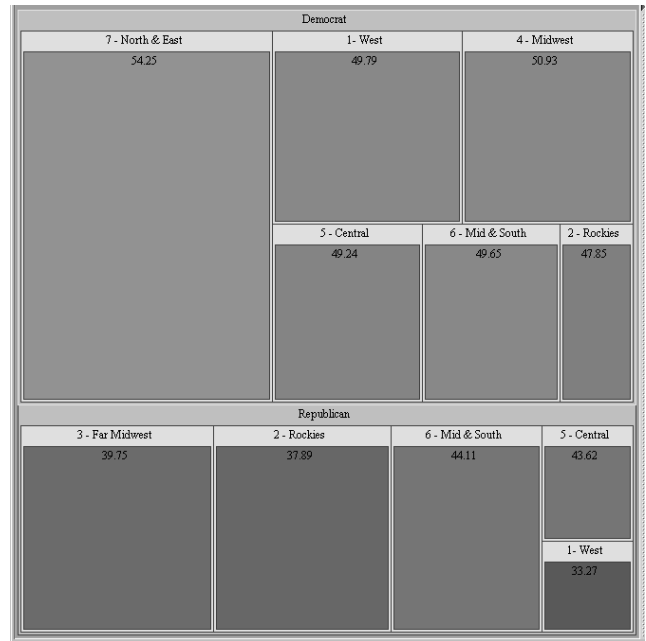


Figure 10b Zoomingrevealsdetailedviewbyshowing additionalhierarchylevelsinthetree

FutureWork

Futureworkmayincludethefollowingtopics:

- **Binningfor nominal valued attributes:** Even if an attribute has nominal values for the analysis it may contain more than needed distinct values. In such a situation, it is desirable to regroup those values so that they will be represented by less number of values. This requires a different approach than the one we presented in this work. The reason is our approach works for values among which there is an order and the alignment on the values axis is fixed. For non ordinal values, a permutation mechanism is needed.
- **Aggregation for nominal valued attributes:** Although we support some statistics for quantitative attributes such a feature for nominal attributes is not available in our system. One has to clarify the meaning of aggregating the nominal values so that they will be presented in the higher levels of hierarchy as a summary.
- **Usability Experiments:** We could not perform many experiments to test the usability of our approach and implementation. A further study may be conducted on experiments to see the impact of the additions introduced in this work, especially the binning mechanism and tour animation.

Conclusion

We have designed and implemented features to support visualization of non-hierarchical data in TreeMap3. The basic concept is a hierarchy editor to allow the user to define hierarchies. A history of hierarchies is stored to enable comparisons with previous visualizations. Tour of hierarchies provides a preview for all the permutations of a selected attribute set. A powerful feature is binning for ordinal and quantitative attributes. The tool lets the user create categories for those attribute values and therefore save the visualization from being unnecessarily detailed. We also support the aggregation for quantitative variables. When only the upper level of the hierarchy is visible, the values in the lower level can be aggregated by a user selected function and this value is propagated to the visible higher-level nodes. We believe the features described in this work are useful and necessary for visualizing non-hierarchical data.

References

- [JS91] Johnson, Brian, and Ben Shneiderman (1991). "Treemaps: A Space-Filling Approach to the Visualization of Hierarchical Information Structures." *Proceedings of IEEE Information Visualization '91*, 275-282.
- [KW01] Erica Kolatch and Weinstein, Beth (2001). "Cat Trees: Dynamic Visualization of Categorical Data"

Using Treemaps. ” *CMSC838B Term Project, May 2001*
http://www.cs.umd.edu/class/spring2001/cmssc838b/Project/Kolatch_Weinstein/

[LR95] Lamping, John, and Rao, Ramana (1995). “The hyperbolic browser: A focus+context technique for visualizing large hierarchies.” *Journal of Visual Languages and Computing* ‘95, 6(4)

[Shn00] Shneiderman, Ben. (1998, 2000). “Treemaps for Space-Constrained Visualization of Hierarchies.”
<http://www.cs.umd.edu/hcil/treemaps/>

[Shn94] Shneiderman, Ben. (1994). “Dynamic Queries for Visual Information Seeking.” *IEEE Software*. 11(6), 70 -77.

[RMC91] Robertson, George G., Mackinlay, Jock, and Card, Stuart (1991). “Conetrees: Animated 3d visualization of hierarchical information.” *Proc. of Computer-Human Interaction '91*, 189 -194.

[Twe97] Tweedie, Lisa (1997). “Characterizing Interactive Externalizations.” *In Proceedings of ACM CHI97 Conference on Human Factors in Computing Systems '97*, 375-382.