Extending the Utility of Treemaps with Flexible Hierarchy

Gouthami Chintalapani[1,3], Catherine Plaisant[1], and Ben Shneiderman[1,2,3]
[1]Institute for Advanced Computer Studies, [2]Department of Computer Science,
[3]Institute for Systems Research
University of Maryland, College Park, Maryland
{plaisant, ben}@cs.umd.edu

Abstract

Treemaps is a visualization technique for presenting hierarchical information on two dimensional displays. Prior implementations limit the visualization to pre-defined static hierarchies. Flexible hierarchy, a new capability of Treemap 4.0, enables users to define various hierarchies through dynamically selecting a series of data attributes so that they can discover patterns, clusters and outliers. This paper describes the design and implementation issues of flexible hierarchy. It then reports on a usability study which led to enhancements to the interface.

Keywords— **Treemap, visualization, flexible hierarchy, graphical user interface, usability study, aggregation, grouping.**

1. Introduction

Effective presentation of information on displays enables users to explore their data which is often hierarchical in nature. The browsing of hierarchies and trees has been investigated extensively [7]. Many visualization techniques use a pre-defined hierarchy to present data sets on computer displays, but exploration can be enhanced if users are given the flexibility to organize the data in various meaningful ways, other than the fixed hierarchy. In this study, we extend treemaps [12] by allowing users to specify the hierarchy shown in treemap. Users can specify categories based on attribute values, visualize the categorized information, interactively manipulate the hierarchy, and save multiple hierarchies.

User specification of the hierarchy is helpful in exploring the relative effects of two or more variables. For example, in election data, a user may wish to see the number of voters in each region followed by information on the gender distribution and finally organized by which party voters are registered in. To understand registration patterns the user may wish to switch to a hierarchy that is organized by party, then gender, then region. For these tasks the user needs a flexible hierarchy.

This paper begins with a brief review of interactive information visualization techniques, followed by a detailed description of design and implementation of flexible hierarchy in the University of Maryland Treemap 4.0 (http://www.hcil.umd.edu/treemap). The next sections describe a usability study conducted to assess the user interface for specifying flexible hierarchies, and reviews interface refinements made to address concerns highlighted by the usability study. The paper concludes with a summary of contributions and possible future extensions.

2. Related Work

Extensive research is being done in the field of interactive visualization of hierarchical data. Several focus+context techniques have been specified to view large hierarchies in their entirety without losing context such as the *hyperbolic tree browser* [15]. The main idea is to lay out the hierarchy uniformly on the hyperbolic plane and map this plane onto a circular display region thereby making effective use of the display. *Reconfigurable disc trees (RDT)* [11], *Magic Eye View* [14], *Hierarchical flip zooming* [4] are other focus+context techniques for visualizing hierarchical information spaces.

Other well-known hierarchy visualization tools include *Cone trees* [18] that present a 3D representation of hierarchical information and enable visualization of the whole structure. The root of the tree is located at the apex of the cone and all its children are arranged around the circular base of the cone in 3D. *Bubble trees* [5] present a tree visualization mechanism based on the natural property of trees to recursively sub-categorize themselves into sub-trees. However, these do not offer a flexible hierarchy to give users control.

Cheops [3] presents another novel approach to the representation, browsing and exploration of huge, complex information hierarchies. The Cheops method is based on compressed visualization of a hierarchical data set and maintains context within a complex hierarchy while providing easy access to details.

Visualization tools designed for data tables can also be used to explore data hierarchically and do make it possible to achieve similar goals to flexible hierarchies. For example during the Infovis 2003 contest (http://infovis.org/infovis2003/) *Infozoom* [21] demonstrated that it could be used effectively to visualize and compare trees. It displays data sets in tables with attributes as rows and objects as columns. Each column shows the leaf of the tree and the path from the leaf to the root is shown by the values in the rows of the table. Users can change the primary, secondary, and tertiary sort orders to achieve results similar to flexible hierarchies.

Tablelens [17] is a focus+context (fisheye) technique, also for visualizing large tables. Tablelens merges symbolic and graphical representations into a single view that can be adjusted by the user. It allows the users to sort and filter data based on values of individual columns. Users can isolate a single variable or group of variable using row focusing techniques and sort successively on those variables thus creating a virtual hierarchy.

Mosaics [9] are space-filling designs composed of contiguous rectangles ("tiles"). Mosaic display is a graphical method for visualizing n-way contingency tables, where the area of the rectangle represents the cell frequencies in the contingency table. The rectangles can be shaded or colored depending on the statistical model used. A collection of related mosaics (also known as mosaic matrix) can be used to show all pair-wise relationships of a set of elements in a multi-way contingency table of categorical variables.

Treemaps [19] were first developed at the Human-Computer Interaction Laboratory (HCIL) of the University of Maryland during the 1990s. Treemaps is a visualization tool that uses 100% of the available display space, mapping two attribute of the data into the size and color of nested rectangular regions. It provides a rapid overview of the relative size of nodes. Dynamic query [1] filters were added in Treemap 3 to facilitate the exploration of data. Users can filter out unwanted items by dragging sliders or selecting values with buttons.

Flexible hierarchy extends the work done in C*atTrees* [13], a University of Maryland class project, which allowed interactive manipulation of a hierarchy. CatTrees was built on early versions of treemap and was limited to categorical data. CatTrees was extended to handle both categorical and numerical data and a new prototype interface for creating and manipulating flexible hierarchies was developed in another class project [20]. The flexible hierarchy features of Treemap 4.0 brings together the explorations of those class projects, and allows users to define new categories based on ranges of attribute values, then define new hierarchies by selecting series of attributes that can be adding to the hierarchy, and save the hierarchies. The interface was refined and integrated in the latest version of Treemap 4.0 available for download from the HCIL website.

3. Design and Implementation

The design and implementation details of the flexible hierarchy user interface are illustrated in this section. Examples presented here use the revised version of the user interface that incorporates the suggestions generated by the usability study.

3.1. Basic treemap functionalities

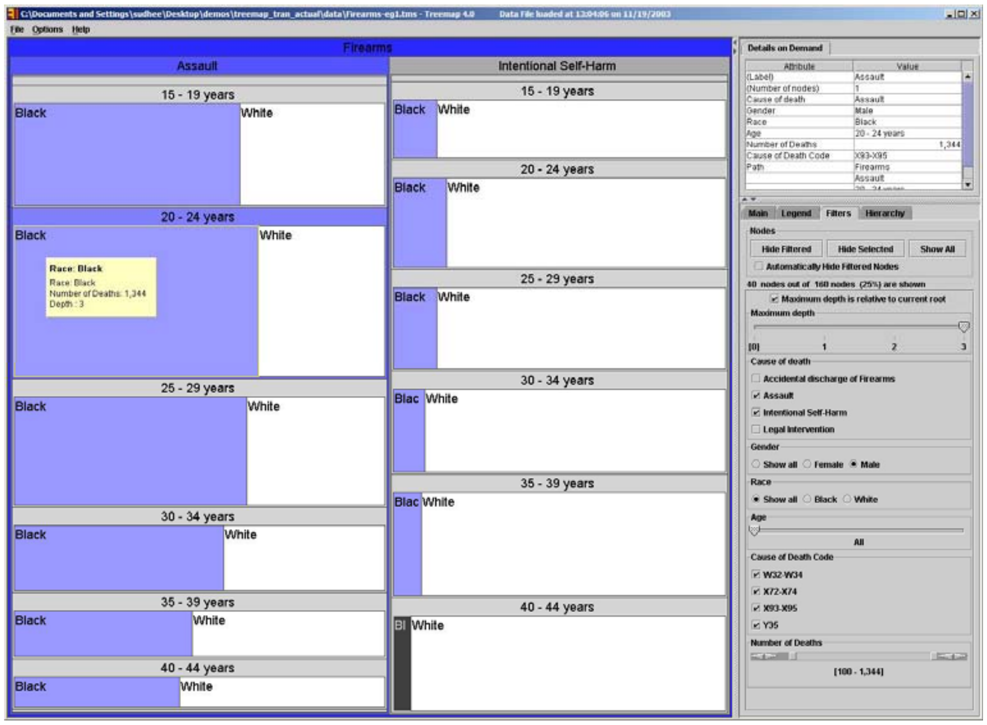
The basic features of treemap are explained using an example showing statistical information about *Firearms* deaths in US (Figure 1). This data is organized by *Cause of death* and *Age* attributes. Each rectangle represents a group of people, identified by their race, age and cause of death. The size of the rectangle is proportional to *Number of deaths* in that group and color represents *Race*, light colored rectangles represent *Whites* and dark colored rectangles represent *Blacks*. Any data attribute can be used for color coding the treemap, while only numerical attributes can be used for sizing.

Comparing the sizes of the rectangles, users can easily spot the groups with highest number of deaths i.e., the largest rectangles (Figure 1). The slice and dice layout, one of the treemap layout algorithms [2], lets the users quickly compare the number of deaths in all groups. Users can see that *Assault* is usually the primary cause of death due to firearms for *Blacks,* and *Intentional self-harm (or suicide)* is the primary cause for *Whites.* The display also highlights that *Assault* is a major cause of death in younger adults.

Clicking on a rectangle displays details in the detail-on-demand window in the top right area of the display and highlights the entire path of that rectangle from the root. As the mouse moves over the treemap the yellow pop-up display shows the values of the attributes assigned to label, size and color. Double-clicking on the border of a group of rectangles zooms in, while right-clicking zooms out level by level.

All the treemap controls are distributed into four tabs: Main, Legend, Filters, and Hierarchy. In the Main tab, users can select any one of the three layout algorithms, squarified, slice and dice, and strip, depending on their needs, as well as font size and border options. The Legend tab allows users to assign attributes to be used for label, size and color options. Treemap enables the users to group the numerical attribute values and assign colors/color gradient to each group. The color-binning widget in the bottom right of Figure 2 shows the histogram distribution of data, lets users specify multiple color gradients.

In the Filter tab, users can filter data using dynamic query sliders [1] as shown on the right bottom window of Figure 1. The filters can be applied to categorical attributes by selecting a value (using item sliders and radio buttons) or by selecting a group of discrete values (using check boxes). The filters for numerical attributes can use double sided range sliders. The items whose attribute values are outside the range are grayed out, and



Attribute	Value
(Label)	Assault
(Number of nodes)	1
Cause of death	Assault
Gender	Male
Race	Black
Age	20 - 24 years
Number of Deaths	1,344
Cause of Death Code	X93-X95
Path	Firearms
	Assault

Nodes: Hide Filtered / Hide Selected / Show All

☐ Automatically Hide Filtered Nodes

40 nodes out of 160 nodes (25%) are shown

☑ Maximum depth is relative to current root

Maximum depth: [0] 1 2 3

Cause of death: ☐ Accidental discharge of Firearms ☑ Assault ☑ Intentional Self-Harm ☐ Legal Intervention

Gender: ○ Show all ○ Female ● Male

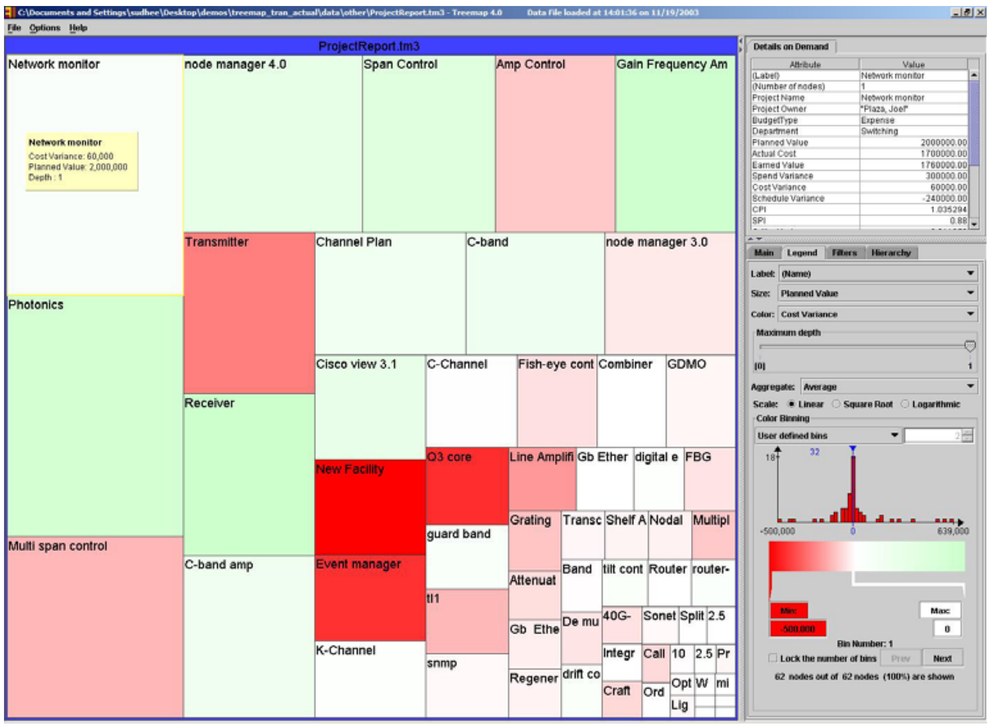
Race: ● Show all ○ Black ○ White

Age: All

Cause of Death Code: ☑ W32-W34 ☑ X72-X74 ☑ X93-X95 ☑ Y35

Number of Deaths: [100 - 1,344]

Figure 1: Visualization of Firearms data, grouped by cause of death and age with slice and dice layout. Size of the rectangle is proportional to the number of deaths and color indicated the race, dark colored rectangles represent African Americans and white colored rectangles represent Whites. Details of the selected node appear on the top right. At the bottom right the filter tab is selected, showing dynamic query buttons and sliders.



Attribute	Value
(Label)	Network monitor
(Number of nodes)	1
Project Name	Network monitor
Project Owner	"Plaza, Joel"
BudgetType	Expense
Department	Switching
Planned Value	2000000.00
Actual Cost	1700000.00
Earned Value	1760000.00
Spend Variance	300000.00
Cost Variance	60000.00
Schedule Variance	-240000.00
CPI	1.035294
SPI	0.88

Label: (Name)

Size: Planned Value

Color: Cost Variance

Maximum depth: [0] 1

Aggregate: Average

Scale: ● Linear ○ Square Root ○ Logarithmic

Color Binning: User defined bins 2

Min: -500,000 Max: 0

Bin Number: 1

☐ Lock the number of bins Prev Next

62 nodes out of 62 nodes (100%) are shown

Figure 2: Overview of 62 projects. No hierarchy is specified. Each rectangle represents a project. Size is proportional to the planned value of the project; Color is proportional to the cost variance. From dark red to pink represents negative cost variance, while green represents positive cost variance.

can be removed from the display with the “hide filtered” button. Customized views can be saved for later use.

3.2. Specifying the hierarchy

Figures 2-5 show a series of visualizations that explain how to create, modify and save various hierarchies, using a different example: project portfolio management data [6]. Each rectangle represents a project. The rectangle size is proportional to the *Planned value* (gross budget) allocated to the project, and color is proportional to *Cost Variance* (budget balance). The dark red colored rectangles indicate over-spent projects, white colored rectangles are almost over-spent, and the light green colored rectangles are under-spent projects.

Figure 2 shows the project portfolio data, without any imposed hierarchical structure. This data is then organized in different ways as shown in Figures 3-5, by imposing meaningful hierarchies on the fly.

Figure 9 shows a close-up view of the user interface for specifying the hierarchy. There are two tables, the hierarchy and attribute tables. Each row shows an attribute name, number of bins and binning type. All available attributes in the data are listed in the attribute table. Users can select an attribute, for example *Project status* and then click on the “add” button to move it to the hierarchy table. This action groups the data by status (Figure 3). Users can select other attributes and add or remove them from the hierarchy, creating a hierarchical structure.

Data is grouped in the order of attributes added to the hierarchy. This order can be changed using “up” and “down” buttons. The “tour” option calculates all the possible permutations of attributes in the hierarchy table and presents the resulting visualizations obtained by these permutations of the attributes.

Users can name the newly created hierarchy, for example *groupedbyStatus* in Figure 3. “New” option lets the users save the current hierarchy and create a new hierarchy. For example, Figure 4 shows a new hierarchical structure imposed by adding the attribute *Department* to the hierarchy. Figure 5 shows the projects grouped by *Region* and then by *Department.* Once the hierarchies are saved, users can switch between different views by selecting them from the pull down menu provided by “Hierarchy List”, top right area of Figure 5.

3.3. Binning the numerical attribute values

In the examples presented so far, hierarchies are created using categorical attributes. Users can also impose hierarchies using numerical attributes. By default, this action results in a crowded visualization with a group (also called bin) for each possible value (Figure 6). This can be avoided by grouping the attribute values into larger bins [9]. Thus users can structure the data by defining meaningful groups for numerical attribute values. For example, using the “% complete” attribute, three equally spaced bins are created: *Initiation Phase* (0-33%), *Implementation Phase* (33-67%) *and Close Out Phase* (67-100%) (Figure 7).

Imposing meaningful groups with numerical data attributes is called binning. Treemap allows three different types of binning, *equally-spaced binning*, *equally-dense binning,* and *user-defined binning*. Data distribution is shown by a histogram; bins are inserted on the histogram using a bin separator, a vertical blue line. The x-axis of the histogram is proportional to the attribute values and the y-axis to the number of rectangles/items with that particular data attribute value.

In equally-spaced binning, the axis of the histogram is divided into equal lengths. Equally-dense binning results in the bins that have same number of items. User-defined binning gives users flexibility to create the bins at any point on the axis. They can move the bin separator with the mouse or by typing-in the value at the bottom of the bin separator, which causes the bin separator to move to the new value. Clicking on the widget creates new bin separators, right clicking on a separator deletes it.

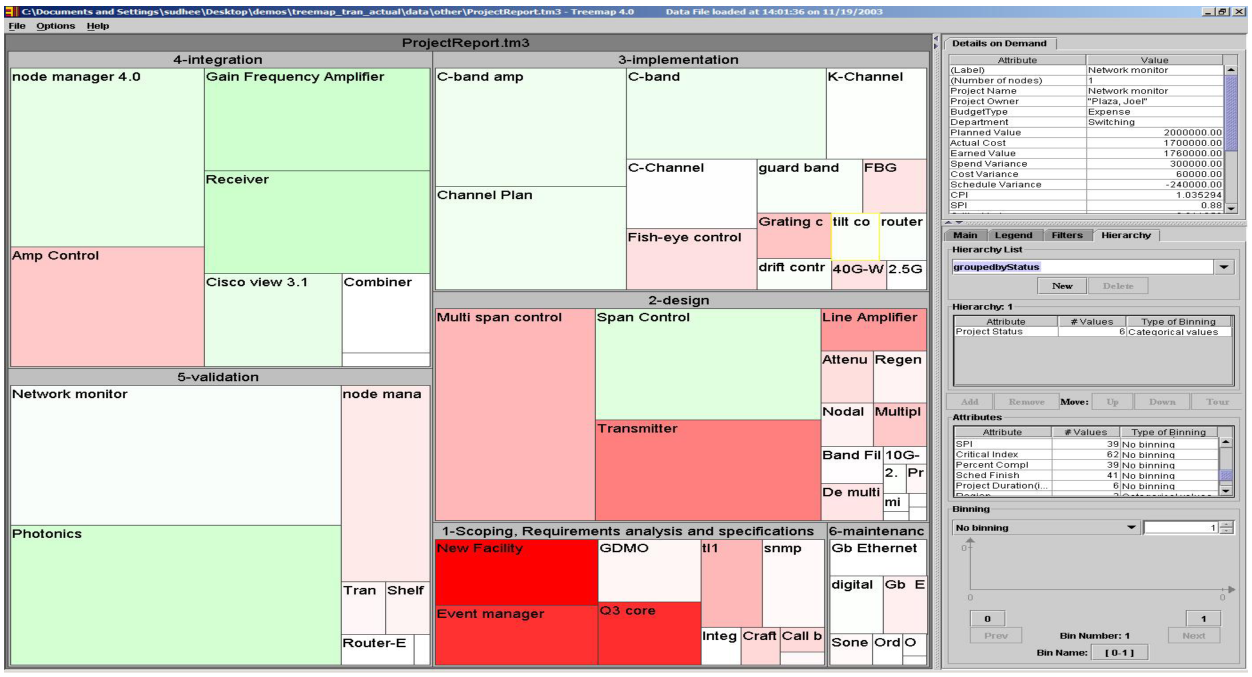
Users can give meaningful names to the bins by typing in a name at the bottom of the binning widget (bin name), for example the bin 67-100% is named as *Close Out Phase* (Figure 9). Users can navigate through the bins by “prev” and “next” options, which is partucularly useful when the bins are small. The information regarding the bin name, number of items in that bin, minimum and maximum numeric values of the bin can be read from the labels provided in the binning widget at the bottom area.

3.4. Aggregation

When the hierarchy is deep and the number of items is large, it might be useful to aggregate the data by controlling the level of detail using the depth slider. An aggregate is a single item that summarizes a group of items below it in the hierarchy [8]. All the rectangles below the chosen hierarchy level are aggregated and represented by a single rectangle. Users can select an aggregate function (average, minimum, maximum and weighted average) for the color attribute. The size of the aggregated rectangle is the sum of the size of the items (by definition in treemap).

3.5. Dealing with imposed variable depth hierarchies

Creating a new hierarchy by selecting a series of attributes always results in a fixed-depth hierarchy. However, designers need often to use imposed fixed hierarchies that are often of variable depth and the data items appear at different levels in the tree structure. Examples of such variable-depth hierarchies are directory structures, disease hierarchies, animal classification, or business organizations. Treemap allows users to combine imposed variable-depth hierarchy and flexible hierarchies. In Figure 10, project portfolio data is grouped by fixed variable depth hierarchy of *location* and then by *Project Status*. Notice that the *East* branch is



C:\Documents and Settings\sudhee\Desktop\demos\treemap_tran_actual\data\other\ProjectReport.tm3 - Treemap 4.0 Data File loaded at 14:01:36 on 11/19/2003

Details on Demand

Attribute	Value
(Label)	Network monitor
(Number of nodes)	1
Project Name	Network monitor
Project Owner	"Plaza, Joel"
BudgetType	Expense
Department	Switching
Planned Value	2000000.00
Actual Cost	1700000.00
Earned Value	1760000.00
Spend Variance	300000.00
Cost Variance	60000.00
Schedule Variance	-240000.00
CPI	1.035294
SPI	0.88

Hierarchy List: groupedbyStatus

Hierarchy: 1

Attribute	# Values	Type of Binning
Project Status	6	Categorical values

Attributes

Attribute	# Values	Type of Binning
SPI	39	No binning
Critical Index	62	No binning
Percent Compl	39	No binning
Sched Finish	41	No binning
Project Duration(i...	6	No binning

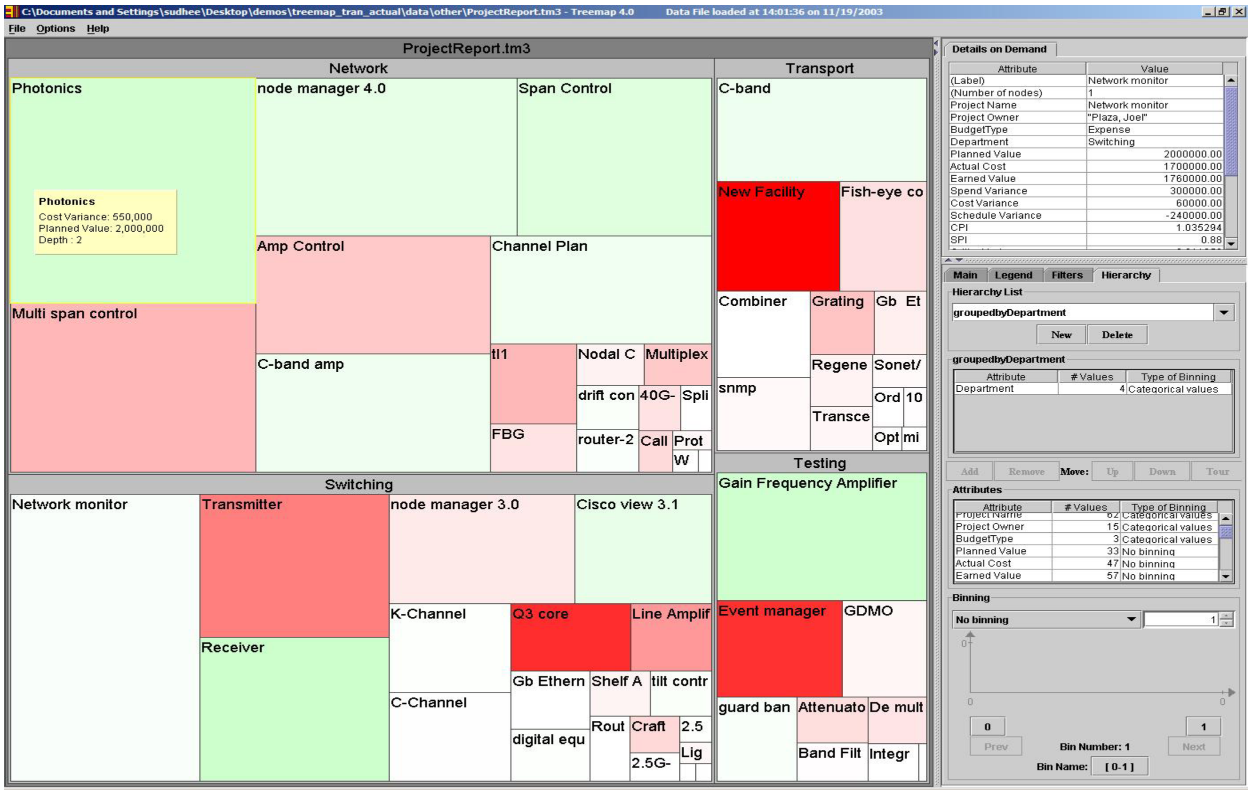
Binning: No binning 1

0 1

Bin Number: 1

Bin Name: [0-1]

Figure 3: Visualization of the same projects, grouped by Project status



C:\Documents and Settings\sudhee\Desktop\demos\treemap_tran_actual\data\other\ProjectReport.tm3 - Treemap 4.0 Data File loaded at 14:01:36 on 11/19/2003

Photonics
Cost Variance: 550,000
Planned Value: 2,000,000
Depth : 2

Details on Demand

Attribute	Value
(Label)	Network monitor
(Number of nodes)	1
Project Name	Network monitor
Project Owner	"Plaza, Joel"
BudgetType	Expense
Department	Switching
Planned Value	2000000.00
Actual Cost	1700000.00
Earned Value	1760000.00
Spend Variance	300000.00
Cost Variance	60000.00
Schedule Variance	-240000.00
CPI	1.035294
SPI	0.88

Hierarchy List: groupedbyDepartment

groupedbyDepartment

Attribute	# Values	Type of Binning
Department	4	Categorical values

Attributes

Attribute	# Values	Type of Binning
Project Name	62	Categorical values
Project Owner	15	Categorical values
BudgetType	3	Categorical values
Planned Value	33	No binning
Actual Cost	47	No binning
Earned Value	57	No binning

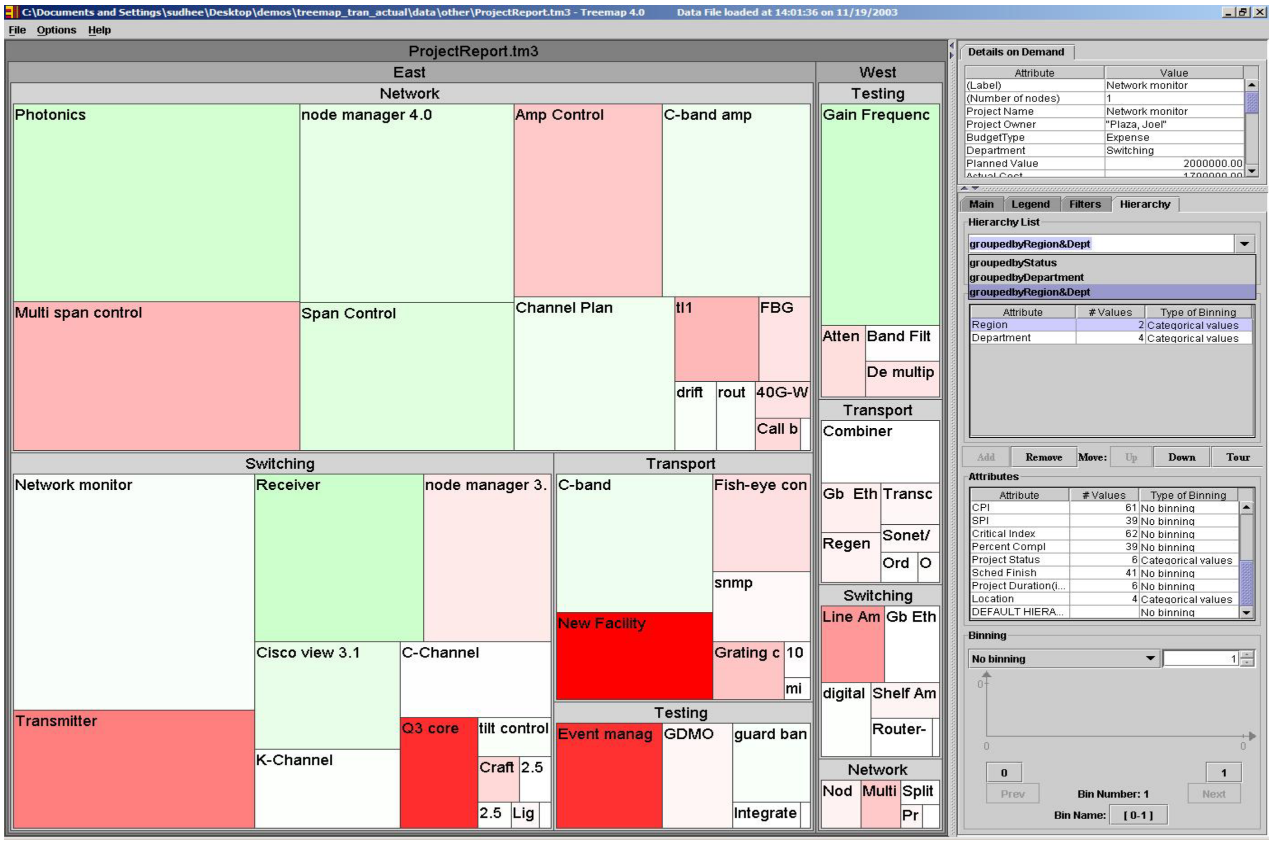
Binning: No binning 1

0 1

Bin Number: 1

Bin Name: [0-1]

Figure 4: Visualization of the same projects grouped by department.



Details on Demand

Attribute	Value
(Label)	Network monitor
(Number of nodes)	1
Project Name	Network monitor
Project Owner	"Plaza, Joel"
BudgetType	Expense
Department	Switching
Planned Value	2000000.00
Actual Cost	1700000.00

Hierarchy List

groupedbyRegion&Dept

groupedbyStatus
groupedbyDepartment
groupedbyRegion&Dept

Attribute	# Values	Type of Binning
Region	2	Categorical values
Department	4	Categorical values

Attributes

Attribute	# Values	Type of Binning
CPI	61	No binning
SPI	39	No binning
Critical Index	62	No binning
Percent Compl	39	No binning
Project Status	6	Categorical values
Sched Finish	41	No binning
Project Duration(i...	6	No binning
Location	4	Categorical values
DEFAULT HIERA...		No binning

Binning

No binning 1

Bin Number: 1

Bin Name: [0-1]

Figure 5: Visualization of the same projects, grouped by region and then by department.

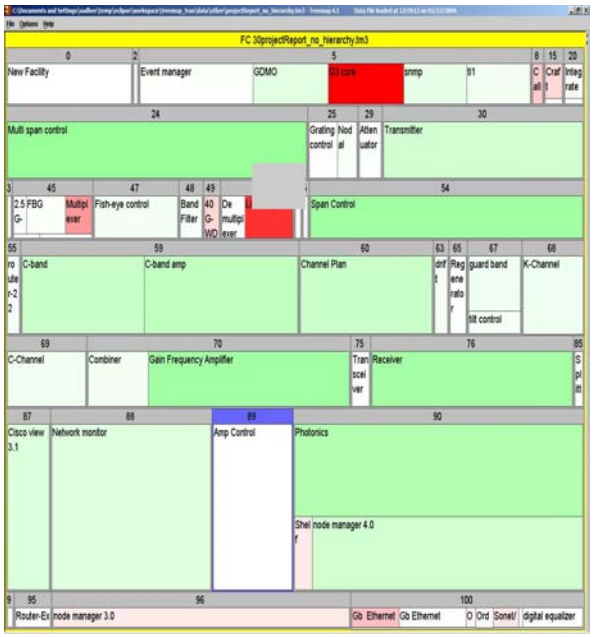


Figure 6: Projects using the strip layout, grouped by percent complete which, without binning, creates a group for each % value.

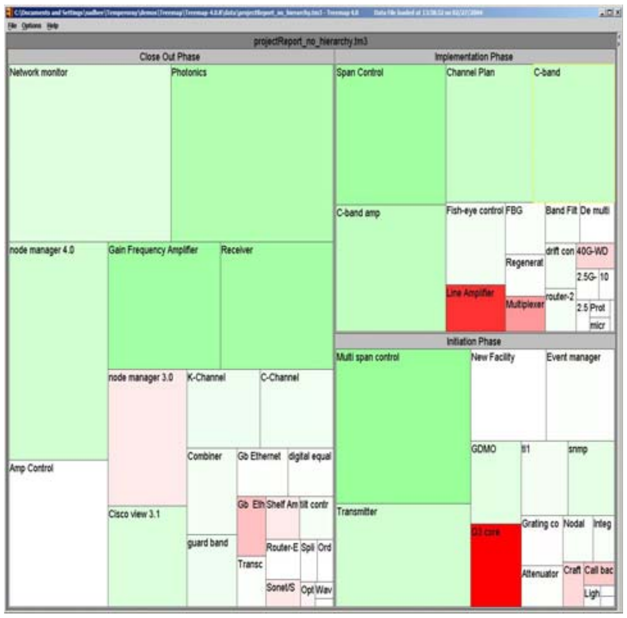


Figure 7: Same as figure 6 but now with three bins created: Initiation Phase (0 – 33%), Implementation Phase (33-67%) and Close Out Phase (67 - 100%).

two levels deep (*Washington DC* at level 2, *Maryland,* and *Virginia* at level 3) while the *West* branch is only one level deep with *San Francisco* at level 2.

Aggregation by hierarchy level may not always be useful for variable depth hierarchies. Instead, aggregating all leaf nodes independently of their depth may be more useful (not yet implemented).

4. Usability study

A usability test was conducted to evaluate Treemap 4.0 with the main emphasis on the flexible hierarchy interface. This test used the original interface (Figure 8), which was then revised based on the test results.

4.1. Participants

The test was conducted with 9 graduate students, of which 4 participants had some experience with earlier treemap versions without flexible hierarchy. The remaining 5 participants had never used the treemap application. All the participants were between 21 and 30 years of age, were computer or civil engineering students, and had extensive experience with computers.

4.2. Data sets used in the test

The data used in the test were about death by firearm statistics and Project management data. The attributes in the Firearms data were Age, Gender, Race, Cause of death and Number of deaths. The attributes in the Project data set were Project name, Project owner, Project status, Percent complete, Scheduled finish date, Department, Region and Location.

4.3. Procedure

Participants were given an introduction to Treemap 4.0 through the narrated recordings of demonstrations of the software, available at http://www.cs.umd.edu/hcil/treemap/doc4.0/toc-video.html) and a short personal demo. Participants who were familiar with treemaps choose not to see the whole video demo. Video demos for individual features such as flexible hierarchy, were available during the test through separate windows, so that participants could refer to them during the test. An introduction about the data was given to the participants before the tasks. The participants were given tasks one by one, and a task was deemed to be complete when the participants were able to answer the question by pointing to the node(s) constituting the answer.

The following tasks were presented to the participants. Each session took about an hour. Early tasks were simple warm-up tasks, while later tasks could benefit from the use of the flexible hierarchy.

Using the *Firearms* dataset:

1. What is the major cause of death?
2. What is the major cause of death in black males?
3. Compare the death rate in males to females in all races and gender.
4. Which age group had the highest death rate, for whites, and for blacks?

Using the *Project report* data set:

1. Find the projects that are out of budget?
2. Identify the project manager responsible for these projects?
3. Group the projects into three groups according to percent complete? The groups are defined as (0-30), (30- 60) and (60-100%).
4. Identify the department to which these projects belong?

4.4. Results

Overall participants were able to understand and create flexible hierarchies, though they had difficulty in understanding the concept of binning numerical attributes.

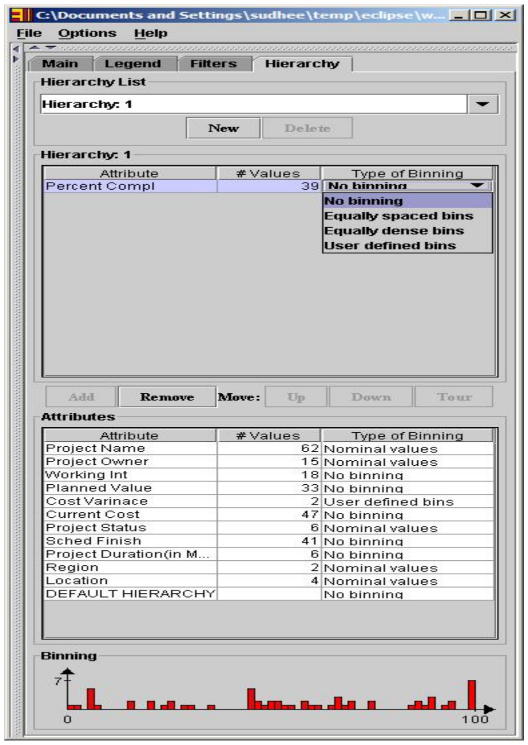
The narrated recordings of demonstrations helped users understand the treemap layout and concept of flexible hierarchy. One of the participants remembered exactly the same sequence of steps that was recorded in the demonstrations and applied them for accomplishing some of the tasks. Only two participants referred to the recorded demos a second time while accomplishing the tasks.

Participants took more time to complete the first 2 to 3 tasks (approximately 1-2 minutes), but as they accomplished more tasks, they were able to proceed with much more ease, and were taking less time (less than 1 minute).

When a task was presented, participants started out exploring the legend tab and tried to accomplish the task using size and color options. They were able to interpret the size and color options easily. They were trying out several options in the “legend tab” to answer the questions. When we suggested that they would use the hierarchy, they were able to produce the desired visualizations. It may be because “legend tab” represents the leaf level attributes, participants might be searching for options to create the tree in the “legend tab” itself. Since both the “legend tab” and “hierarchy tab” deal with the presentation of information, the participants may have assumed they must be closely related in some manner.

Participants were able to create a one level hierarchy and interpret the visualization by identifying the groups formed as a result of imposing the hierarchy. We observed participants adding and removing attributes multiple times to understand the result. They were able to identify the groups and the subgroups easily for two level hierarchies but seem to have more difficulty for 3 or more levels.

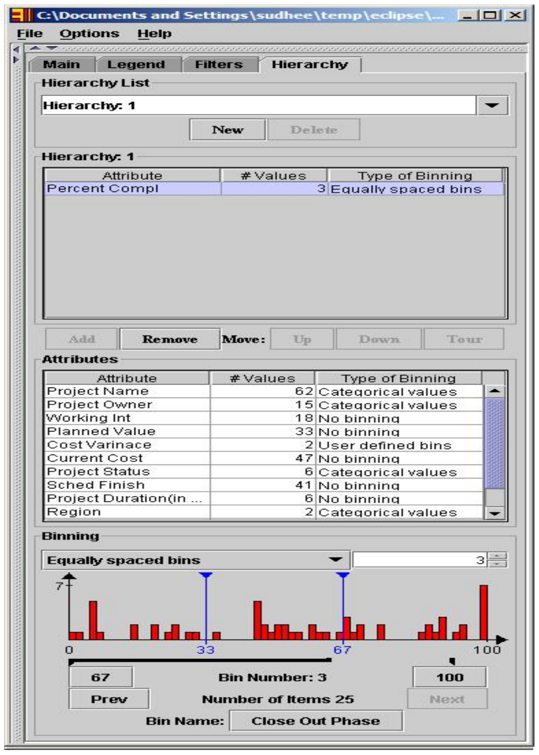
Participants were able to modify the hierarchies by adding different attributes and removing the ones they didn’t need. Most of the participants did not use the “move up” or “move down” options to change the order



Attribute	# Values	Type of Binning
Percent Compl	39	No binning

Attribute	# Values	Type of Binning
Project Name	62	Nominal values
Project Owner	15	Nominal values
Working Int	18	No binning
Planned Value	33	No binning
Cost Varinace	2	User defined bins
Current Cost	47	No binning
Project Status	6	Nominal values
Sched Finish	41	No binning
Project Duration(in M...	6	No binning
Region	2	Nominal values
Location	4	Nominal values
DEFAULT HIERARCHY		No binning

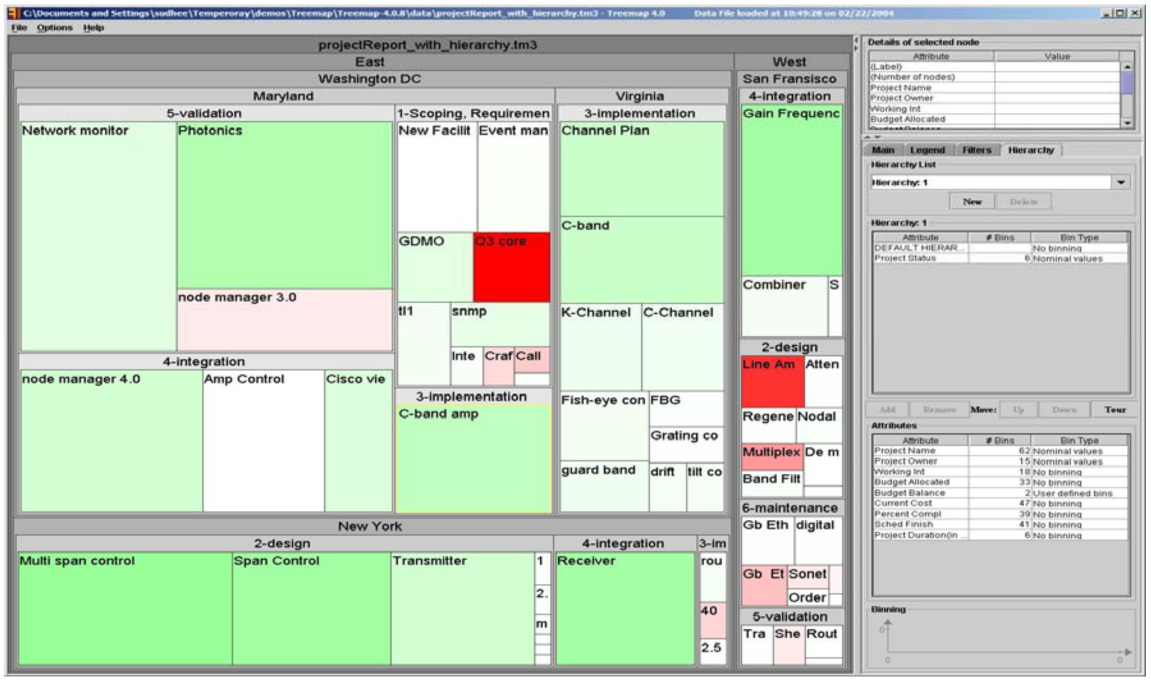
Figure 8: Original Implementation of Flexible Hierarchy user interface



Attribute	# Values	Type of Binning
Percent Compl	3	Equally spaced bins

Attribute	# Values	Type of Binning
Project Name	62	Categorical values
Project Owner	15	Categorical values
Working Int	18	No binning
Planned Value	33	No binning
Cost Varinace	2	User defined bins
Current Cost	47	No binning
Project Status	6	Categorical values
Sched Finish	41	No binning
Project Duration(in ...	6	No binning
Region	2	Categorical values

Figure 9: Revised Flexible Hierarchy user interface showing the Hierarchy table, Attributes table, and binning widget.



Attribute	Value
(Label)	
(Number of nodes)	
Project Name	
Project Owner	
Working Int	
Budget Allocated	

Attribute	# Bins	Bin Type
DEFAULT HIERAR...		No binning
Project Status	6	Nominal values

Attribute	# Bins	Bin Type
Project Name	62	Nominal values
Project Owner	15	Nominal values
Working Int	18	No binning
Budget Allocated	33	No binning
Budget Balance	2	User defined bins
Current Cost	47	No binning
Percent Compl	39	No binning
Sched Finish	41	No binning
Project Duration(in ...	6	No binning

Figure 10: Visualization of same projects, grouped first using the fixed variable-depth hierarchy of region and location, and then by project status.

of hierarchy. Instead they removed and added the attributes. None of the participants tried the “tour” option. Also, for the tasks pertaining to the same data set, participants were asked to save the hierarchy before proceeding to another task. Treemap allows users to save multiple hierarchies in the same treemap file, but users did not use that feature and saved separate files for each hierarchy.

Participants had difficulty understanding the terminology used in binning, for example the terms like *#bins*, *nominal values* and *no binning*. While answering task 3 using *Project report* data set, participants had problems in using binning. They selected the *percent complete* attribute and added it to the hierarchy, which resulted in a crowded visualization with one group per attribute value. Only one participant figured out the binning options. Two other participants successfully referred to the recorded demonstrations, and the rest had to be given a hint.

Specifying the bins was difficult in part because several widgets presented inadequate affordances (i.e. there were insufficient cues as to the behavior of the widgets). For example to see the binning types for the numerical attributes users had to click on the third column of the attribute/hierarchy table, which would reveal the pull down menu. In equally-spaced and equally-dense bins, double clicking the second column of the numerical attribute would let the users specify the number of bins. While creating bins, the color of the bin separator (green) was obstructing users from reading the value. Users adjusted the bin position using mouse-drag actions but it was frustrating as they were not able to position the bin exactly where they wanted to. There was no good hint in the interface that the bin position could be changed by clicking on the separator value and typing a new value in the box. One of the participants interpreted the label, showing the number of items in a bin, to be the range of the bin. For equally-spaced and equally-dense bins, participants were not able to add and remove the bins. While removing the bins in user-defined binning, participants ended up in creating more bins. Any mouse-click (either left or right) would create a bin, where as a right mouse-click on the arrow of bin separator would delete that bin.

The study suggests that binning numerical attributes is a complex feature and users need more training to use it. This feature had been requested by our advanced treemap users and implemented to allow any complex binning, but remains a challenge for novices. Our experience suggests that the benefits of these features become clear only when users start analyzing their own data, i.e. data they care about.

5. Modifying the interface based on usability study

The problems found in the usability study were addressed and the following changes were made to original interface, leading to improved labeling, a revised binning editor and new options.

The headers for the attribute table and the hierarchy table were changed, from #bins to #Values, indicating that the attribute has x number of distinct values and results in x bins when added to the hierarchy, “Binning Type” to “Type of Binning”, and “Nominal values” to “Categorical values”. More labels were added to the bin editor. When the bins are very close, attribute values overlap and are hard to see. In the revised interface, bin values are displayed on the buttons in the bottom of Figure 9.

The number of items was removed from the histogram itself, placed below the histogram and labeled “Number of Items”. A label showing the bin number was also added to facilitate navigation among the bins. Users can navigate the bins using “Prev” and “Next” options. Two lines were drawn from the histogram to the buttons showing the values. In “user-defined binning”, bins are created with a left mouse-click and deleted with the right mouse-click on the arrow. This prevents users from accidentally creating bins when they are actually deleting them.

The bin editor of the revised version (Figure 9) was made consistent with the one in the “legend tab” (used for color binning the numerical attributes), thus making it easy to use both bin editors. When the users select the numerical attributes in either the attribute or the hierarchy table, the bin editor is highlighted. This highlighting depending on the attribute drives the attention of the users and prompts them to explore the bin editor. Additional methods were provided to edit the bins. Users can change the binning type by selecting an option in the drop down menu provided by the combo box. Number of bins can be changed with the bin counter.

Each bin is identified by its numeric range, for example [0-33], in treemap. When the numeric values are big (more than 5 digits), labels in the treemap may not be easily read. Instead, users can now assign a meaningful name to the numeric ranges. For example, the bin [67-100] is named as *Close Out phase* in Figure 9. These names can be saved along with the other settings.

The data attributes can be added and removed from the attribute and hierarchy tables using mouse drag and drop actions. This would be helpful for advanced users. While rearranging the attributes in a hierarchy, users can select the attribute, drag and drop it at the new position.

6. Conclusions

A new interface to dynamically create and manipulate hierarchies has been implemented. This extension to treemaps enables users to impose hierarchies with a series of attributes, to modify and save the hierarchies, and to switch between different views. Binning and aggregation play an important role in visualizing the data, providing detail to the extent preferred by the users. Users can categorize the data using binning, filter the unwanted data with dynamic query filters and aggregate the lower level information.

Improved aggregation which enables users to aggregate all the leaf nodes at once, tour of various hierarchies, binning by node (forming three groups, with less than, equal and greater than that node's value) coupling data items in treemap with histogram display (clicking on a node would show its position in the histogram and clicking on a bar in the histogram would highlight items in treemap) are few possible future directions to improve the tool.

Acknowledgements We thank Evren Sirin, Fusun Yaman, Jean-Daniel Fekete, Niem H Dang, Ani Jain, and Aleks Aris for their support in the implementation of flexible hierarchy and Hyunmo Kang for preparing the recorded demonstrations. We thank Chevron-Texaco for their support of this work. We also appreciate partial support from National Science Foundation grants for Information Technology Research (#0086143) "Understanding the Social Impact of the Internet: A Multifaceted Multidisciplinary Approach" and Digital Government Initiative grant (EIA 0129978) "Towards a Statistical Knowledge Network."

References

[1] Ahlberg, C., Williamson, C., and Shneiderman, B., Dynamic queries for information exploration: An implementation and evaluation. *Proc. SIGCHI: Human Factors in Computing Systems, 1992.* ACM, 619–626. May 1992.

[2] Bederson, B.B., Shneiderman, B., and Wattenberg, M., Ordered and quantum treemaps: making effective use of 2D space to display hierarchies. *ACM Transactions on Graphics, 2002.* ACM, 833-854. October 2002.

[3] Beaudoin, L., Parent, M-A., Vroomen, L.C., Cheops: a compact explorer for complex hierarchies. *Proc. 7th Conference on Visualization 1996.* IEEE, 87-ff. October 1996.

[4] Bjork, S., Hierarchical flip zooming: enabling parallel exploration of hierarchical visualizations. *Proc. Working Conference on Advanced Visual Interfaces, 2000.* IEEE, 232-237. May 2000.

[5] Boardman, R., Bubble Trees: Visualization of hierarchical information structures. *Chi '00 Extended Abstracts on Human Factors in Computer Systems, POSTER SESSION: Student Posters, 2000.* ACM, 315-316. April 2000.

[6] Cable, J., Ordonez, X., Chintalapani, G., Plaisant, C., Project portfolio earned value management using treemaps, to appear *Proc. Project Management Institute Research conference PMI2004*, July 11-14, London, UK (http://www.pmi.org).

[7] Card, S. K., Mackinlay, J. D., Shneiderman, B., *Readings in Information Visualization: Using Vision to Think.* Morgan Kaufmann Publishers. 1999.

[8] Fredrikson, A., North C., Plaisant, C., and Shneiderman, B., Geographical and categorical aggregations viewed through coordinated displays: a case study with highway incident data. *Proc. Workshop on New Paradigms in Information Visualization and Manipulation, 1999.* ACM, 26-34. November 1999.

[9] Friendly, M., A brief history of the mosaic display. *Journal of Computational & Graphical Statistics, 2002.* Volume 11, Number 1, 89-107, March 2002.

[10] Goldstein, J., Roth, S., Using aggregation and dynamic queries for exploring large data sets. *Proc. SIGCHI Conference on Human Factors in Computing Systems, 1994.* ACM, *23-29.* 1994.

[11] Jeong, C., Pang, A., Reconfigurable disc trees for visualizing large hierarchical information space. *Proc. of Information Visualization, 1998*. IEEE, 19-25. 1998.

[12] Johnson, B., Shneiderman, B., Treemaps: a space-filling approach to the visualization of hierarchical information structures. *Proc. 2nd International Visualization Conference 1991.* IEEE, 284-291. October 1991.

[13] Kolatch, E., Weinstein, B., CatTrees: Dynamic visualization of categorical data using Treemaps. *url:* http://www.cs.umd.edu/class/spring2001/cmsc838b/project/Kolatch_Weinstein, 2001.

[14] Kreuseler, M., and Schumann, H., A flexible approach for visual data mining. *IEEE Transactions on Visualization and Computer Graphics, Special selections on Information Visualization and Visual Data Mining 2002.* IEEE, Vol. 8, No. 1, 39-51. January – March 2002.

[15] Lamping, J., Rao, R., Pirolli, P., A focus+context technique based on hyperbolic geometry for visualizing large hierarchies. *Proc. SIGCHI Conference on Human Factors in Computing Systems 1995.* ACM, 401-408. May 1995.

[16] Plaisant, C., Chintalapani, G., Lukehart, C., Schiro, D. and Ryan, J., Using visualization tools to gain insight into your data. *Proc. Society of Petroleum Engineering Annual Technical Conference, 2003.* Denver, CO. October 2003.

[17] Rao, R., Card, S., The table lens: merging graphical and symbolic representation in an interactive focus+context visualization for tabular information. *Proc. SIGCHI Conference on Human Factors in Computing Systems, 1994*. ACM, 318-322. April 1994.

[18] Robertson, G., Mackinlay, J., Card, S., Cone trees: animated 3D visualizations of hierarchical information. *Proc. SIGCHI Conference on Human Factors in Computing Systems 1991*. ACM, 189-194. March 1991.

[19] Shneiderman B., Tree visualization with tree-maps: A 2-D space-filling approach. *ACM Transactions on Graphics, 1992.* ACM, 92-99. January 1992.

[20] Sirin, E., Yaman, F. Visualizing dynamic hierarchies in Treemaps. url: http://www.cs.umd.edu/class/spring2002/cmsc838f/project/, 2002.

[21] Spenke, M., Beiken, C., Visualization of trees as highly compressed tables with InfoZoom, *Poster compendium of IEEE Symposium on Information Visualization, 2003.* IEEE 2003.