Structure-Based Brushes: A Mechanism for Navigating Hierarchically Organized Data and Information Spaces *

Ying-Huey Fua, Matthew O. Ward and Elke A. Rundensteiner
Computer Science Department
Worcester Polytechnic Institute
Worcester, MA 01609
{yingfua,matt,rundenst}@cs.wpi.edu

Abstract

Interactive selection is a critical component in exploratory visualization, allowing users to isolate subsets of the displayed information for highlighting, deleting, analysis, or focussed investigation. Brushing, a popular method for implementing the selection process, has traditionally been performed in either *screen space* or *data space*. In this paper, we introduce an alternate, and potentially powerful, mode of selection that we term *structure-based brushing*, for selection in data sets with natural or imposed structure. Our initial implementation has focussed on hierarchically structured data, specifically very large multivariate data sets structured via hierarchical clustering and partitioning algorithms. The structure-based brush allows users to navigate hierarchies by specifying focal extents and level-of-detail on a visual representation of the structure. Proximity-based coloring, which maps similar colors to data that are closely related within the structure, helps convey both structural relationships and anomalies. We describe the design and implementation of our structure-based brushing tool. We also validate its usefulness using two distinct hierarchical visualization techniques, namely hierarchical parallel coordinates and tree-maps. Finally, we discuss relationships between different classes of brushes and identify methods by which structure-based brushing could be extended to alternate data structures.

Keywords: Brushing, hierarchical representation, interactive selection, exploratory data analysis.

*This work is supported under the NSF grant IIS-9732897 and the NSF CISE Instrumentation grant #IRIS 97-29878.

1 Introduction

It has long been recognized that a hierarchical organization on a dataset provides a nature framework for viewing information at varying levels-of-detail. In recent years , there has been increased interest in visual exploration of such hierarchies [13]. Many commercial applications, such as Microsoft Windows Explorer and Norton Commander, provide visualizations of hierarchically structured information. However, a major disadvantage of these somewhat simplistic visual presentations is that there is a limited display space for the tree. Hence, it is easy to lose context, for instance, if some subdirectories get very large.

To address this issue, there have been numerous research efforts toward finding effective methods to present and explore hierarchical information, such as Tree-Maps [8, 14], Cone-Trees [12] and Reconfigurable Disc Trees [7]. However, most of these methods provide only limited modes of interactions for navigating the hierarchy. In this paper, the particular way of representing the tree structure is not of concern; rather we are interested in finding ways to effectively navigate through such hierarchies. Navigation plays an important role in aiding users to find their way through the complex structure: to see where they are, what information is available, and how to access information of interest, while avoiding getting lost wandering in some isolated subspace.

Often we are interested in selecting and exploring a particular subspace of interest after having an overview of a data set. One popular way of achieving such selection is through *brushing*. Brushing is an interactive process for selecting subsets of data or localizing a subspace within an N-dimensional space [10, 16, 19]. Such interactive selection then provides users with the ability to highlight, delete, mask, analyze, aggregate, or focus investigation on a small, manageable portion of the data set. Brushing consists of painting sections of

the display using a mouse or stylus to indicate the data elements to be selected. Traditionally, brushing has been performed in either a *display-driven* or *data-driven* mode. In the former, a connected region of the display canvas is indicated, and all data falling within the subregion are marked as selected, while in the latter, subranges of the data dimensions are specified to define a subset of interest.

In this paper, we introduce an alternate, and potentially powerful, mode of selection that we term *structure-based brushing*, that can be used to perform selection in data sets with natural or imposed structure. Our initial implementation has focussed on hierarchically structured data, specifically very large multivariate data sets organized via hierarchical clustering and partitioning algorithms [2, 3]. By extending the brush metaphor to structure, our *structure-based brush* allows for navigation of hierarchical space by specifying focal extents and level-of-detail on a visual representation of the structure. In particular, this new brush serves as a mechanism to select subsets of the hierarchical structure for further analysis or drill-down/roll-up operations. We also use *proximity-based coloring*, which maps similar colors to data that are closely related within structure space, to help convey both structural relationships and anomalies. Specifically, we propose and analyze several alternatives for realizing structure-based coloring schemes.

In this paper, we also describe the design and implementation of our structure-based brushing tool. We then illustrate the usefulness and generality of our tool by applying it to two distinct hierarchical visualization techniques, namely, hierarchical parallel coordinates [2] and tree-maps [8, 14]. The case studies we report have been conducted using a real data set, consisting of a 5-dimensional 16,000 element dataset formed by combining SPOT, magnetics, and radiometrics (three channels) remote sensing datasets from the Grant's Patch region of Western Australia. Finally, we discuss relationships between dif-

ferent classes of brushes and identify methods by which structure-based brushing could be extended to alternate data structures such as lists or graphs.

While we initially introduced the basic idea of structured-based brushing in a symposium paper [3], we now go beyond this work in several ways. In particular, besides more detailed discussions, we now also (1) develop a categorization of *proximity-based coloring* schemes for both binary as well as n-ary hierarchical structures, (2) describe the design and implementation of our structure-based brushing tool and its application to several hierarchical visualization techniques, (3) study relationships between different classes of brushes, and lastly (4) identify methods by which structure-based brushing could be extended to apply to other types of structures.

Section 2 of this paper gives an overview of the basics of brushing. Section 3 introduces the concept of structure-based brushes. Section 4 describes the creation and manipulation of our new brush, and Section 5 contains details of our structure-based coloring scheme. Following that in Section 6, we present two case studies, namely hierarchical parallel coordinates and tree-maps. Thereafter, we identify relationships between different classes of brushes in Section 7 and describe how our techniques can be applied to other types of structures in Section 8. We conclude by summarizing our contributions and outlining plans for future work.

2 Brush Basics

Selection is a process whereby a subset of entities on a display are isolated for further operations, such as highlighting, deleting, or analysis. Wills [18] defined a taxonomy of

selection operations, classifying techniques based on whether memory of previous selections is maintained or not, whether the selection is controlled by the underlying data (we term this *data-driven*) or not, and what specific interactive tool (e.g., brushing, lassoing) is used to differentiate an area of the display. He also created a selection calculus that enumerates all possible combinations of actions between a previous selection and a new selection (replace, add, subtract, intersect, and toggle) and attempted to identify configurations of these actions that would be most useful.

Brushing is the process of interactively painting over a subregion of the data display using a mouse, stylus, or other input device that enables the specification of location attributes. It has been used as a method for performing selection in graphics for many years. The principles of brushing were first explored by Becker and Cleveland [1] and applied to high dimensional scatterplots. In this system, the user specified a rectangular region in one of the 2D scatterplot projections and, based on the mode of operation, points in other views corresponding to those falling within the brush were highlighted, deleted or labelled. Ward and Martin [16, 10] extended brushing to permit brushes to have the same dimensionality as the data (N-D instead of 2-D). They also explored the concepts of multiple brushes, composite brushes (formed by logical combinations of brushes), and *fuzzy* brushes, which allow points to be partially contained within a brush. Haslett et al. [5] introduced the ability to show the average value of the points that are currently selected by the brush.

One common method of classifying brushing techniques is by identifying in which space the selection is being performed, namely screen or data space. This can then be used to specify a *containment criteria* (whether a particular point is inside or outside the brush). In *screen space* techniques, a brush is completely specified by a 2-D contiguous subspace on the screen. In *data space* techniques, a complete specification consists of either an enumeration of the data elements contained within the brush or the N-D boundaries of

a hyperbox that encapsulates the selection. In this paper, we introduce a third category, namely *structure space* techniques, where selection is based on structural relationships between data points such as trees or lists.

In addition, brush manipulation may be *direct* or *indirect*. Direct manipulation refers to the ability to interactively control brush creation and manipulation by mouse (or other locator) actions on the data display itself. On the other hand, indirect manipulation refers to the use of separate widgets such as sliders to specify or manipulate the brush coverage. Direct manipulation is generally preferred for data-driven operations (such as isolating an interesting subset of the display), while user-driven operations (such as a range query) are often easier to specify with indirect methods.

3 Structure-Based Brushes

The *structure* of a data set specifies relationships between data points. This structure may be explicit (e.g., categorical groupings or time-based orderings) or implicit (e.g., resulting from analytic clustering or partitioning algorithms). Examples of structures include linear orderings, hierarchies, and directed acyclic graphs. In this work, we focus on hierarchical relationships of data, as it is one of the modeling principles that is receiving substantial attention as a means for effectively coping with large data sets [3, 8, 14, 12, 7].

A *hierarchy* or *tree* is a convenient mechanism for organizing large data sets. By recursively clustering or partitioning data into related groups and identifying suitable representative information (summarizations) for each cluster, we can examine the data set methodically at different levels of abstraction, moving down the hierarchy (*drill-down*)

when interesting features appear in the summarizations and up the hierarchy (*roll-up*) after sufficient information has been gleaned from a particular subtree.

There has been much research into the graphical presentation of hierarchical or tree-structured information [7, 8, 12, 14]. However, most techniques for interacting with the resulting displays have focused on modifying the viewing parameters or distorting the structure to help identify features of interest. We have created a suite of techniques, which we term *structure-based brushing*, aimed at supporting the interactive exploration of large data sets that are either implicitly or explicitly organized into a hierarchical structure.

As mentioned in Section 2, brushing requires a containment criteria. For our first containment criteria, we augment each node in the hierarchy, that is each cluster, with a monotonic value relative to its parent. This value can be, for example, the level number, the cluster size or population, or the volume of the cluster's extents (defined by the minimum and maximum values of the nodes in the cluster). This assigned value determines the control for the level-of-detail. By choosing a continuous control variable, such as cluster size, the traversal of the tree through different levels-of-detail can be smooth transitions instead of abrupt screen changes. In addition, this concept can be extended to allow the specification of a subset of levels-of-detail. Hence, instead of selecting a single level of detail, we might want to select multiple levels-of-detail (which we term the *level-of-detail range* or, for short, LOD range). Formally, we define a level-of-detail variable L, with range l_{min} to l_{max} and an LOD range $[l_a, l_b]$ such that $l_{min} \leq l_a \leq l_b \leq l_{max}$.

Our second containment criteria for structure-based brushing is based on the fact that each node in a tree has extents, denoted by the left- and right-most leaf nodes originating from the node. In addition, it is always possible to draw a vertically oriented tree in such a way that the horizontal position of each node (and, in fact, all of its children) falls between

its extents. These extents ensure that a selected subspace is contiguous in structure space. Formally, we define an extents variable E, with range e_{min} to e_{max} and an extent range $[e_c, e_d]$ such that $e_{min} \leq e_c \leq e_d \leq e_{max}$.

Thus a structure-based brush is defined by a subrange of the structure extents and level-of-detail variables, namely $[l_a, l_b, e_c, e_d]$.

4 Creation and Manipulation of Structure-Based Brush

4.1 The Structure-Based Brushing Interface

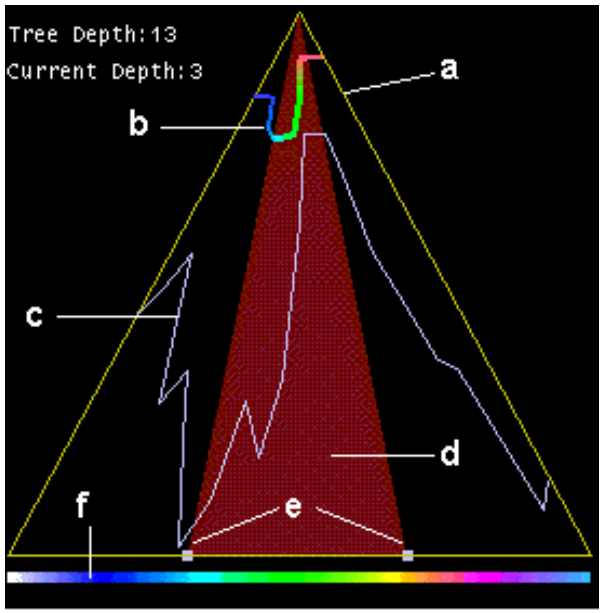

Figure 1: Structure-based brushing tool. (a) Hierarchical tree frame; (b) Contour corresponding to current level-of-detail; (c) Leaf contour approximates shape of hierarchical tree; (d) Structure-based brush; (e) Interactive brush handles; (f) Colormap legend for level-of-detail contour.

Figure 1 shows our *structure-based brushing interface*. The triangular frame depicts

the hierarchical tree (see (a)). The leaf contour (see (c)) depicts the silhouette of the hierarchical tree. It delineates the approximate shape formed by chaining the leaf nodes. The colored bold contour (see (b)) across the tree delineates the tree cut that represents the cluster partition corresponding to the specified level-of-detail. The color ramp (f) below the triangle indicates the colors that will be assigned to different sections of the hierarchy. The same colors are used for the display of the nodes in the corresponding data display. The two movable handles (see (e)) on the base of the triangle, together with the apex of the triangle, form a wedge in the hierarchical space (see (d)).

Each node is assigned a color via a process we've termed *proximity-based coloring* [2]. Proximity is based on the structure of the hierarchical tree, that is, sibling nodes are considered closer than non-sibling nodes. We first impose a linear order on the data clusters gathered for display at a given LOD value, w. This is done in a recursive top-down manner using an in-order tree traversal. Finally, we assign colors to each cluster by looking up a linear colormap table. Details of this algorithm are presented in the next section (Section 5).

4.2 Brush Manipulation

4.2.1 Using the Structure-Based Brush for Range Selection

Our structure-based brushing tool supports both direct and indirect manipulation. Sets of elements may be directly selected by positioning the wedge handles so as to bound the range of colors spanned by the elements. This is made possible due to the direct color correspondence between the data display and the structure display. Moreover, similar elements are selected as a group, since, by our coloring criteria, similar elements are drawn in similar colors. The wedge handles can be adjusted at either end or the existing brush may

be simply translated to bound the desired set of elements. Indirect manipulation is provided through the use of sliders for the range of extents and values, in case the user prefers this mode of interaction.

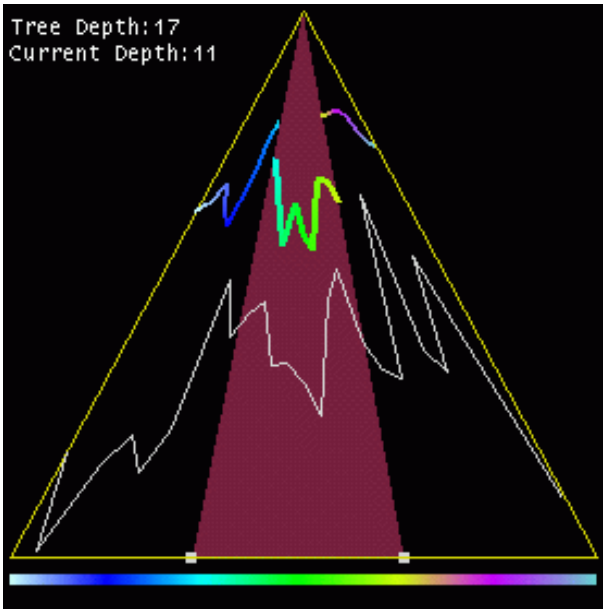

Figure 2: Structure-based brushing at two different levels-of-detail.

4.2.2 Drill-down and Roll-up Operations

In a hierarchical organization, drill-down and roll-up operations are commonly used to explore the hierarchy. Our tool supports a global drill-down and roll-up operation, that is, the current level-of-detail can be adjusted by dragging the colored contour vertically. The data display changes to reflect more detail when the contour is adjusted vertically downwards, while showing more and more abstract views of the data when the contour is adjusted vertically upwards.

Besides a global drill-down/roll-up operation, our tool also allows the user independent control of the level-of-detail of the brushed and unbrushed region. That is, the colored

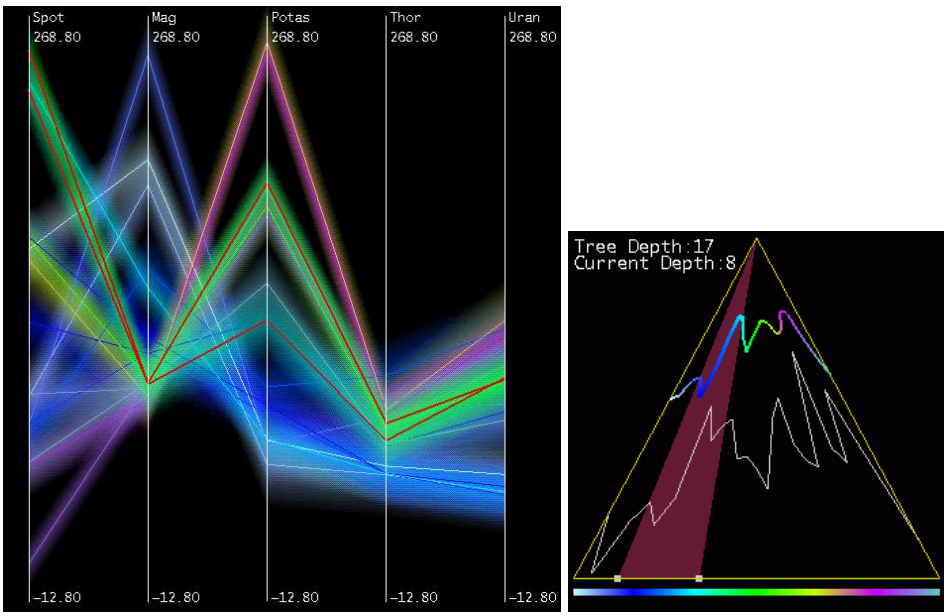

Figure 3: A hierarchical parallel coordinates display of a remote sensing dataset with the selected cluster painted in bold red to reflect that it is currently being brushed in the structure-based tool. The image on the right shows the corresponding level-of-detail indicated by the colored contour in the structure-based brush, with the brushed region indicated by the wedge. In this case, we observe that the selected clusters share the same mean value for magnetics and uranium contents, and have high SPOT contents.

contour in the brushed region can be adjusted independently of the contour segments outside the brushed region, and vice versa (See example in Figure 2). We term this *selective drill-down/roll-up*. This separate mode of control gives the user the flexibility to view the hierarchical structure at two different levels-of-detail at the same time.

4.2.3 Semantics of Hierarchy Navigation Operations

The semantics of drill-down and roll-up operations bear some scrutiny. The drill-down operation appears to have a single interpretation associated with it, namely select all children of all previously selected non-terminal nodes. For the exploration task we extend this to include any terminal node in the original selection, as these nodes are at the lowest level of detail. We have identified three distinct forms for the roll-up operation, named ANY, MOST, and ALL modes. The ANY semantic states that a parent node is selected if any of its children were in the previous selection set. The ALL semantic, as the name implies, only selects a parent node if all of its children were in the previous selection. Roll-up with the MOST semantic implies that a parent is selected only if at least half of its children were selected before. The reasoning behind this option is that the color assigned to a parent node, as described in Section 5, is approximately midway between the extents of its children's colors. Database query processing strategies for each of the operations above are described in detail in [15].

5 Conveying Structure with Color

Color can be used to reinforce structural relationships between nodes in a hierarchy and convey correspondences between the structure and data displays. Ideally we want a color-

ing strategy that has the following properties:

- sibling nodes have nearly the same color (thus the same proximity-based coloring),
- a parent node has a color within the range of its children's colors so that familial relations are clear,
- the color space is effectively utilized, i.e., there are no significant parts of the color space to which no node is assigned, and
- differences in color between non-sibling nodes are readily discernible compared to the difference between siblings.

We are investigating a number of different algorithms for proximity-based coloring for hierarchically structured data. The specific structure of the tree in terms of the branching configuration determines the complexity of the applicable algorithms.

For the following algorithms, we treat color as a scalar variable. In our current implementation, we use $C()$ to indicate the hue component of an HSV colormap. Alternate color maps are possible as well. The color value assigned to each node T_i of the tree is denoted by $C(T_i) \in [0, 1]$, i.e., it is a normalized color value. T_0 is the root of the tree, and $C(T_0)$ then the color of the root. For a binary tree we can assign colors to nodes of the hierarchy based on the following recursive formula:

$$\begin{aligned} C(T_0) &= 0.5 \\ C(T_i) &= C(parent(T_i)) + \frac{\pi(i)}{K^{l_i+1}} \end{aligned} \tag{1}$$

where K is the branching-factor of the cluster tree, l_i is the tree depth at node T_i, and $\pi(i)$ is the sign function defined as:

$$\pi(i) \quad = \quad \begin{cases} +1 & \text{if i is odd} \\ -1 & \text{if i is even} \end{cases} \tag{2}$$

This equation does not differentiate between adjacent elements (with respect to the linear order) belonging to different subtrees. It is important to distinguish between such elements because such adjacent elements are deemed "significantly separated" according to our proximity measure. For this, we revise Equation (1) by introducing a "buffer" between subtrees. The buffer acts as an unused color interval between subtrees so that elements at the proximal ends of subtrees are not assigned colors that are indistinguishable. Clearly the buffer should be larger between large clusters and smaller otherwise.

Let b, where $b < 1$, be the desired buffer interval. Let the revised definition be:

$$C(T_i) \quad = \quad C(parent(T_i)) + \pi(i) \left(b^{l_i} + \frac{1}{K^{l_i+1}} \right) \tag{3}$$

Equation (3) achieves our desired purpose. We typically choose b to be small with values around 10^{-1}.

For non-binary trees we are exploring three distinct approaches, as briefly described below.

Fixed Branch Factor: If the given tree structure can be assigned a fixed maximum branching factor for all its nodes, we can readily modify Equation 1 to place sibling nodes closer together, with positions alternating between the left and right sides of the parent and emanating outward to the full range of the color subspace assigned to the parent. This, however, can lead to significant wastage of the color space, especially

if most nodes do not have the maximum branching factor. Gaps can be inserted between adjacent non-sibling nodes by reserving a small percentage of the range available at one end of the color values assigned to a node.

Single Look-Ahead: This method divides the range of colors associated with a node equally amongst its children. This means that the distance between siblings of different parents on the same level will not necessarily be the same. There can be some wastage of the color space, but not as much as with the Fixed Branch Factor. Gaps can be easily incorporated in the algorithm for separating subtrees.

Population-Based: This bottom-up approach assigns colors evenly among all terminal nodes (with gaps added between adjacent non-sibling nodes), and then assigns the parent the average color of its children. Gaps can easily be handled by this technique by simply inserting space at the end of each sibling set. This provides the best utilization of the color space of the three approaches.

The Population-Based method seems to meet our evaluation criteria the best among the three approaches. It has however one significant drawback; in situations where the tree undergoes incremental changes (adding/deleting nodes or subtrees), the entire tree must be relabeled. In the Fixed Branch Factor method, there is some room for addition, up to the maximum branch count, while the Single Look-Ahead method can easily accommodate changes locally by redistributing siblings over the existing range (which then propagates down to their offspring).

Proximity-based coloring highlights the relationships among clusters. It is however not always possible to impose a linear order on the data clusters. For instance, a cluster chain forming a circular loop is not amenable to any linear order. In this case, an arbitrary break

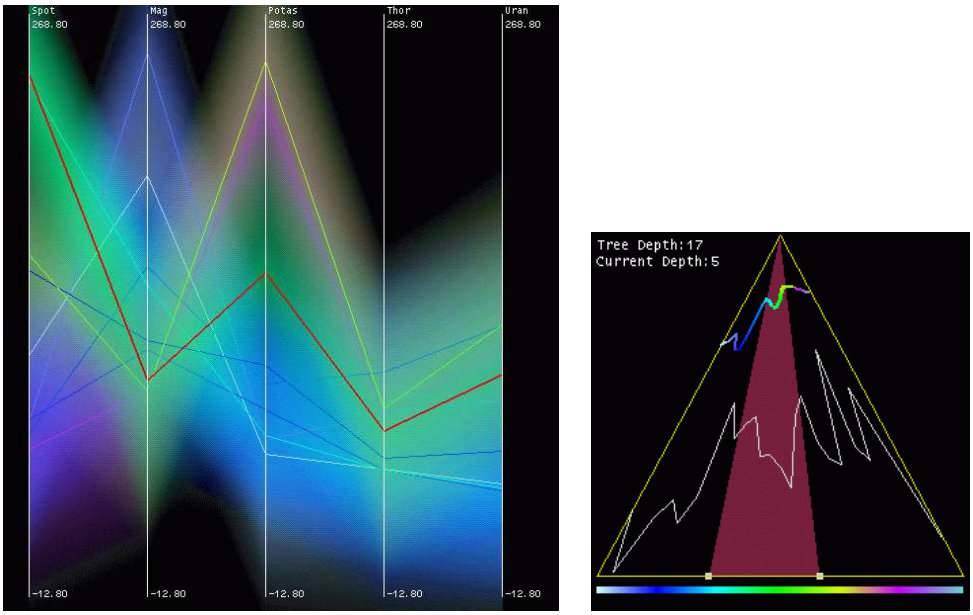

Figure 4: A hierarchical parallel coordinates display of a remote sensing data with the selected and unselected clusters at the same level-of-detail. Notice that the selected cluster that is drawn in bold red has relatively low mean levels of magnetics and thorium contents. The colored contour in the structure-based brush indicates the current level-of-detail.

must be made at some point in the loop. Data elements at the break point, though similar according to our proximity measure, would be assigned contrasting colors.

6 Case Studies

We illustrate the usefulness and general applicability of our tool by applying it to two hierarchical visualization techniques: hierarchical parallel coordinates [2] and tree-maps [8]. These case studies demonstrate the functionality of our new brush, its usefulness, and difference from alternative techniques.

We use a 5-dimensional 16,000 element dataset formed by combining SPOT, magnetics, and radiometrics (three channels) remote sensing datasets from the Grant's Patch region of

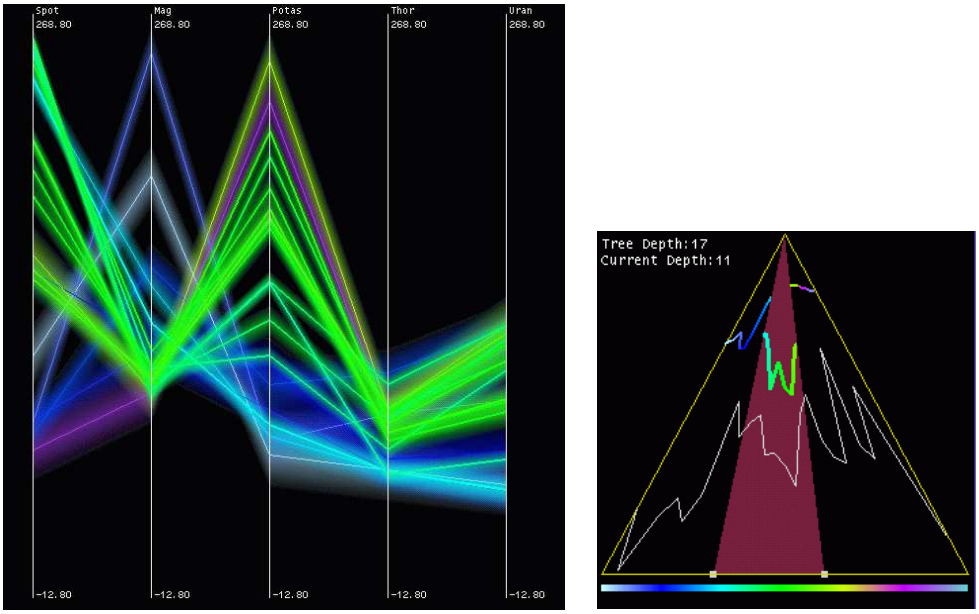

Figure 5: A hierarchical parallel coordinates display of a remote sensing dataset with the selected cluster drawn at a higher level-of-detail compared to the unselected clusters. The left image shows the effect on the selected cluster (indicated by the bold red lines in Figure 4) when it expands to show more detail. In this case, we display the original colors of the selected lines rather than painting them bold red in order to reveal the actual colors encoded for the clusters. Moreover, in order to display the lines clearer, we reduce the bands around the lines via extent scaling [2]. These clusters exhibit trends similar to their parent cluster, that is, having relatively low mean levels of magnetics and thorium contents. The corresponding levels-of-detail are indicated by the structure-based brush on the right.

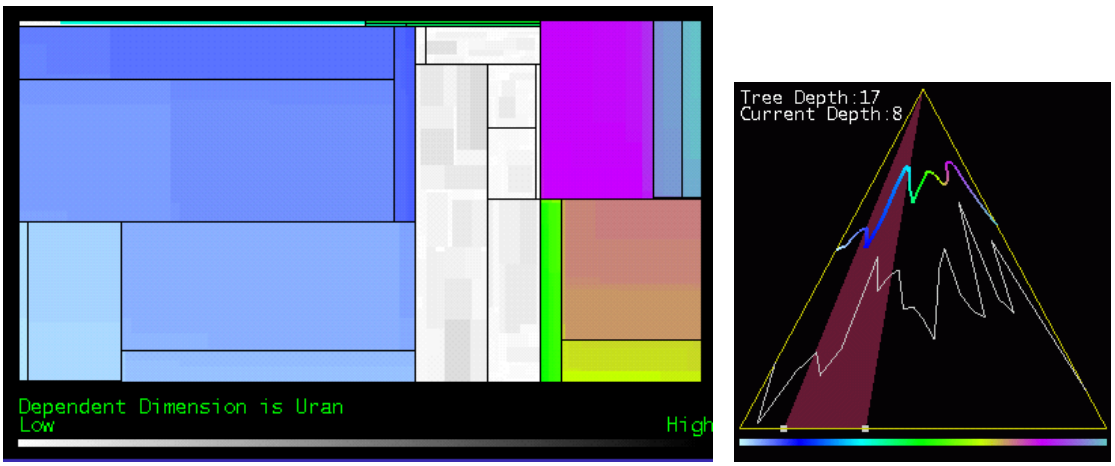

Figure 6: A tree-map display of the remote sensing dataset with the selected clusters painted with the color of the dependent variable, uranium. By mapping the value of the dependent variable to a greyscale colormap where high values are mapped to darker colors, we observe that the selected clusters have relatively low mean levels of uranium content.

Western Australia. The hierarchical clustering was achieved by processing the data with the BIRCH algorithm [20], which can handle large scale data sets efficiently.

6.1 Interacting with Hierarchical Parallel Coordinates using Structure-Based Brushes

Parallel coordinates is a multivariate visualization technique pioneered in the 1980's that has been applied to a diverse set of problems [6, 17]. In this technique, each data dimension of an N-dimensional data set is represented as a (horizontal or) vertical axis, and the N axes are organized as uniformly spaced lines. A data element in an N-dimensional space is mapped to a polyline that traverses all of the axes, intersecting each axis at a position proportional to its value for that dimension.

Hierarchical parallel coordinates [2] is a new extension that we have developed for visualizing large multivariate data sets. In hierarchical parallel coordinates, the data is struc-

tured as a hierarchy of clusters, and the display shows summarizations of the clusters at a certain level of detail. Many display options exist, including showing cluster centers (which look identical to traditional parallel coordinate displays), extents (which manifest themselves as variable width bands encasing the centers), population (mapping to opacity of the extent bands), and other cluster statistics. Distortion techniques, proximity-based coloring, and selective fade-in/fade-out are available to help reduce clutter and expose structure.

Figure 3 shows a hierarchical parallel coordinates display at a given level-of-detail. Each polyline across the axes displays the mean value of its cluster. The number of polylines spanning the screen corresponds to the number of clusters at the given level-of-detail. The lines in the data display are painted with the corresponding color of the structure-based display, with the color red reserved as a highlighting color. With our brushing tool, the user simply adjusts the handles at the base of the triangle wedge to bound the extents of interest. The selected clusters are drawn in bold red, indicating they are being brushed. Next, we demonstrate the usefulness of the selective drill-down/roll-up operations.

Figures 4 and 5 show two images of hierarchical parallel coordinates at different levels-of-detail. Figure 4 displays the initial state, with all clusters at the same level-of-detail. The user can then brush the cluster(s) of interest by adjusting the handles at the bottom of the wedge on the structure-based brush interface. Next, by "pulling" the brushed contour vertically downwards, we can view the selected clusters at a higher level-of-detail while maintaining the same level-of-detail for the unselected clusters. This results in the display shown in Figure 5. We have turned off the red encoding of the brushed clusters to convey the actual colors of the clusters that correspond to the colored contour. The usefulness of the selective drill-down/roll-up feature is evident here; users have the flexibility to see an isolated view or to manipulate the region of interest while minimizing the distraction from data lines not falling in that region.

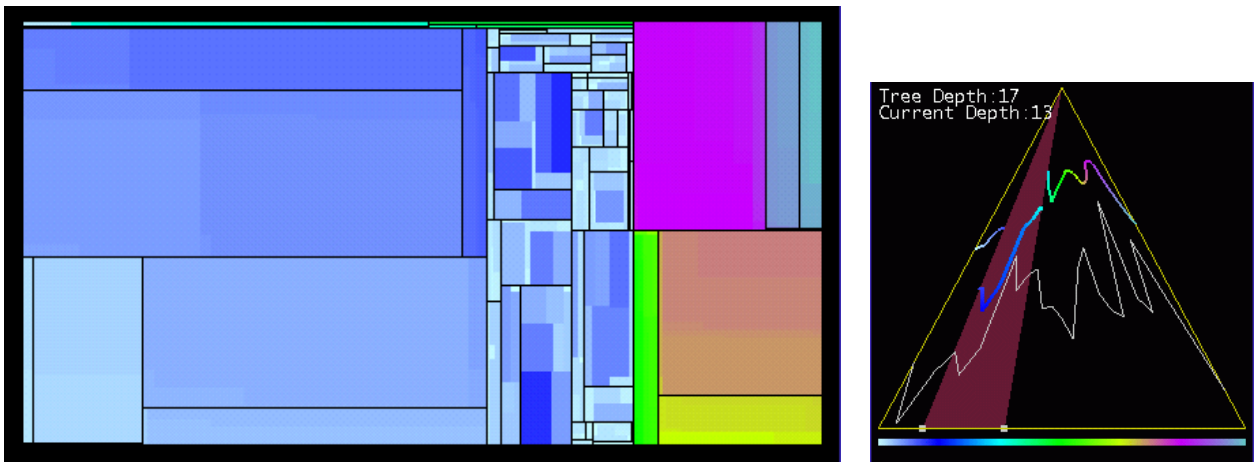

Figure 7: A tree-map display of a remote sensing dataset with the selected clusters at a higher level-of-detail compared to the unselected clusters. In this case, we color the selected rectangular regions with the corresponding color on the structure display. The structure-based brush is shown on the right.

6.2 Interacting with Tree-Maps using Structure-Based Brushes

A tree-map [8, 14] is a space-filling method for presenting hierarchical, univariate data. It is formed by taking a rectangular display area and recursively subdividing it based on the tree structure, alternating between horizontal and vertical subdivisions and allocating area proportional to the number of terminal nodes in each subtree. The terminal rectangular regions are filled with a color based on the dependent variable. In our modified version of the tree-map, we can choose to fill the color of the rectangular region based on a dependent variable or with its corresponding color from the structure display (See Section 5 on proximity-based coloring).

Figure 6 shows the display of a tree-map at a given level-of-detail as indicated by the colored contour in the structure-based brush. As in the hierarchical parallel coordinates display (Section 6.1), the clusters of interest can be selected by bounding the corresponding color on the structure-based brush interface. The color of the selected clusters on the tree-

map then change to reflect the value of the dependent variable of the tree-map using a grey-scale color ramp.

Next, the user can view the selected cluster on the tree-map at a higher level of detail by "pulling down" the brushed segment of the colored contour in the structure-based brush. The resulting state of the structure-based brush and the cluster display are shown in Figure 7. In this case, we choose to paint the nodes using proximity-based coloring. We can observe the relative sizes of each subcluster from further subdivisions of the rectangle. Since all these observations are isolated from the unselected clusters, it gives the user an uncluttered view of the regions of interest. To make similar observations for other clusters, the users can simply translate the wedge if they desire the same brush size, or adjust the handles at the corners of the wedge to define a totally new brush.

7 Relationships Between Classes of Brushes

It is important to differentiate structure-based brushing from traditional data-based brushing. In a traditional user-driven brushing operation, to specify a region of interest in a multivariate data display, the user sets upper and lower bounds for each dimension. In data-driven brushing, the user paints over groupings of interesting data. Neither of these approaches is suitable for isolating data elements that are *structurally* related. Rather, their focus is on the *values* of the data. Clearly, structure-based brushing provides new, and potentially invaluable, functionality beyond data-based brushing.

A logical question is whether it is possible and, if so, useful to switch between structure-based and data-driven brushes. Each is a specialization of what we term *set-based brushing*, where every point is classified as either being within the brush or outside the brush. In

addition, as we've implemented with ramped brushes [10], each point can have a degree of membership, associated with, for example, its distance from the focus relative to the extents.

In explicit hierarchies, which are often generated via categorical data or ranges of data values, there is generally a direct mapping from structure-based brushes to data driven brushes, but not vice versa. A user could specify a subspace of the hierarchy that maps to a contiguous subspace in data space. This would correspond directly to a data-driven brush. However, a slight enlargement of one of the data dimension extents could correspond to adding a disjoint partial branch of the hierarchy to the selection. In other words, there is no direct translation back from the data-driven brush to the structure-based brush semantics. The user would then have to decide if the modifications to the structure-based brush being specified by the data-driven brush would follow an ANY, MOST, or ALL semantic for defining the containment criterion (see discussion in Section 4). One potential means of addressing the issue of disjoint segments of the structure being selected could possibly be the reorganization of the branches of the tree, assuming the order of branches is not semantically significant.

For implicit hierarchies and explicit hierarchies not derived by subranging of the dimensions, the relationship is weaker, since any branch of the hierarchy can contain points that are disjoint in data space. The degree to which this disjointness occurs depends on the quality of the clustering process in use. Indeed, it would be a useful procedure to compare different clustering algorithms by examining how well the points within the structure cluster in the data display. Similarly, data points within a small region of data space can be located throughout the structure space. This might be useful in exploring the distribution of different classes or characteristics of data over a hierarchy to identify imbalances.

A more detailed analysis of these conversions between brush types will be the topic of a future investigation.

8 Brushing Other Structures

The concept and methods of structure-based brushing can be extended beyond hierarchies to other data structures. For example, regular and irregular grids, as well as arbitrary graphs, can be considered for structuring objects in an information space. The main components of the brush would be the focal point and extents. Navigation would be equivalent to positioning the focus, while the size of the region of interest would be specified by modifying the extents. One additional parameter that we have found useful is a density specification, which indicates what percentage and distribution of the data within the extents and surrounding the focus should be displayed.

In the hierarchical tool we've presented in this paper, the focus corresponds to the line between the root and terminal node at the center of the brush. Extents are controlled via the handles specifying the breadth of the data of interest, and the density corresponds to the level of the hierarchy to be shown.

In grid structures (e.g., 1-D, 2-D, or 3-D field data), focus is specified as a single point within the grid, extents define the subspace of interest as in a range query, and density indicates how many points are to be displayed. This can be denser at locations closest to the focus and sparse near the boundaries of the extents. We have used this concept for interactive selection of subsets of data from fire simulations generated via computational fluid dynamics [11].

For arbitrary graphs, the focus again specifies a single node in the information space. The notion of extents depends on the types of the links between nodes. For an unweighted bidirectional graph, extents can indicate the number of links from the focus that define the subspace of interest. For directional graphs, this would be amended to allow either uni-directional extents or bi-directional extents. For weighted graphs, the extents could be specified as a threshold, and only nodes connected to the center with a strength greater than the threshold (either directly or via products of strengths along a path emanating from the node) would be displayed. The concept of density in this case is less intuitive than the other structures, but could indicate a sampling of the points within the extents.

In all cases, techniques such as distortion [4, 9], fading, and clipping can be used to deemphasize the parts of the information space not included in the selection. This is useful for maintaining context while decreasing the clutter. As in brushing, these filtering operations can be done in screen space, data space, or structure space.

9 Conclusions

This paper presented a new technique for navigating hierarchies, called structure-based brushing, that is an extension of the data-based brush metaphor. With structure-based brushing, it is possible to select a subset of a hierarchy and explore the selected space in varying levels-of-detail using drill-down/roll-up operations.

The concept of *proximity-based coloring* was also introduced for mapping similar colors to data that are closely related within structure space. This process helps convey both structural relationships and anomalies. Specifically, we proposed and analyzed several

alternative methods for realizing hierarchical coloring schemes. We also explored relationships between different classes of brushes, and identified methods by which structure-based brushing could be extended to apply to other types of structures. In this paper, we also described the design and implementation of our structure-based brushing tool we have developed as part of the XmdvTool project [3, 2]. We then illustrated the usefulness and generality of structure-based brushing by integrating it with two distinct hierarchical visualization techniques, namely, hierarchical parallel coordinates [2] and tree-maps [8, 14]. Other examples of its use, as well as source code for XmdvTool, can be found at our project website located at http://davis.wpi.edu/∼xmdv.

There are several limitations to our current structure-based brushing tool. First, the extent-based subranging assumes that the order of the branches is fixed. With a different order of the clusters, the color assignment will be different, and hence the selection. Also, with our coloring strategy, adjacent clusters may be assigned indistinguishable colors if the number of clusters is very large.

Our future work will be aimed at addressing these limitations among other related tasks. In particular, we are interested in using zooming/distortion techniques in both structure space and color space to facilitate precise operations on dense structures. We are also planning to investigate methods for dynamically reordering cluster branches (when ordering isn't data-driven) to more readily enable the comparison of multiple isolated branches. While this could be accomplished using multiple composite brushes [10], dynamic reorganization may lead to simpler exploratory interactions.

References

[1] A. Becker and S. Cleveland. Brushing scatterplots. *Technometrics, Vol 29(2), p. 127-142*, 1987.

[2] Y. Fua, M. Ward, and E. Rundensteiner. Hierarchical parallel coordinates for exploration of large datasets. *Proc. of Visualization '99, p. 43-50*, Oct. 1999.

[3] Y. Fua, M. Ward, and E. Rundensteiner. Navigating hierarchies with structure-based brushes. *Proc. of Information Visualization '99, p. 58-64*, Oct. 1999.

[4] G. Furnas. Generalized fisheye views. *Proc. of Computer-Human Interaction '86, p. 16-23*, 1986.

[5] J. Haslett, R. Bradley, P. Craig, A. Unwin, and G. Wills. Dynamic graphics for exploring spatial data with application to locating global and local anomalies. *Statistical Computing 45(3), p. 234-42*, 1991.

[6] A. Inselberg and B. Dimsdale. Parallel coordinates: A tool for visualizing multidimensional geometry. *Proc. of Visualization '90, p. 361-78*, 1990.

[7] C. Jeong and A. Pang. Reconfigurable disc trees for visualizing large hierarchical information space. *Proc. of Information Visualization '98, p. 19-25*, 1998.

[8] B. Johnson and B. Shneiderman. Tree maps: A space-filling approach to the visualization of hierarchical information structures. *Proc. of Visualization '91, p.284-91*, 1991.

[9] Y. Leung and M. Apperley. A review and taxonomy of distortion-oriented presentation techniques. *ACM Transactions on Computer-Human Interaction Vol. 1(2), p. 126-160*, 1994.

[10] A. Martin and M. Ward. High dimensional brushing for interactive exploration of multivariate data. *Proc. of Visualization '95, p. 271-8*, 1995.

[11] R. J. Resnick, M. O. Ward, and E. A. Rundensteiner. Fed - a framework for iterative data selection in exploratory visualization. *Proc. of SSDBM '98, p. 180-189*, 1998.

[12] G. Robertson, J. Mackinlay, and S. Card. Cone trees: Animated 3d visualization of hierarchical information. *Proc. of Computer-Human Interaction '91, p. 189-194*, 1991.

[13] D. Schaffer, Z. Zuo, S. Greenberg, L. Bartram, J. Dill, S. Dubs, and M. Roseman. Navigating hierarchically clustered networks through fisheye and full-zoom methods. *ACM Transactions on Computer-Human Interaction, Vol. 3(2), p. 162-88*, 1996.

[14] B. Shneiderman. Tree visualization with tree-maps: A 2d space-filling approach. *ACM Transactions on Graphics, Vol. 11(1), p. 92-99*, Jan. 1992.

[15] D. I. Stroe, E. A. Rundensteiner, and M. O. Ward. Minmax trees: efficient relational operation support for hierarchical data exploration. *WPI Technical Report WPI-CS-TR-99-37*, 1999.

[16] M. Ward. Xmdvtool: Integrating multiple methods for visualizing multivariate data. *Proc. of Visualization '94, p. 326-33*, 1994.

[17] E. Wegman. Hyperdimensional data analysis using parallel coordinates. *Journal of the American Statistical Association, Vol. 411(85), p. 664*, 1990.

[18] G. Wills. Selection:524,288 ways to say this is interesting. *Proc. of Information Visualization '96, p. 54-9*, 1996.

[19] P. Wong and R. Bergeron. Multiresolution multidimensional wavelet brushing. *Proc. of Visualization '96, p. 141-8*, 1996.

[20] T. Zhang, R. Ramakrishnan, and M. Livny. Birch: an efficient data clustering method for very large databases. *SIGMOD Record, vol.25(2), p. 103-14*, June 1996.