Multiform Views of Multiple Trees

Martin Graham & Jessie Kennedy
Napier University, Edinburgh, UK
{m.graham@napier.ac.uk, j.kennedy@napier.ac.uk}

Abstract

We describe a case study of TaxVis, a multiple view system for examining relationships between sets of multiple classification trees. The system displays multiform views of the dataset, which in turn can either be a singular view of the larger forest object formed from the set of trees, or multiple views in themselves, using linking to show relationships between the separate trees. We describe the circumstances that led to the development of the multiple view aspects of the application and how it emerged that different views were indeed suited to different tasks.

Keywords--- **Multiform Visualization, Multiple Trees, Linking, Brushing**

1. Introduction

The TaxVis project has the aim of developing a visual application that allows users to explore the relationships between multiple taxonomies. Development of consequent visualizations has lead to a realization that there is no one singular form of visualization that can best support all the proposed tasks that users want to perform. In this paper we describe how different types of visualization best answer different questions and thus why it is necessary to use multiform and multiple views.

We describe related work in the area, followed by a description of our application, including the various view types that are used along with details of their interaction and organization. We then discuss the issues that led to the development of the application as a multiple view system

2. Related Work

Multiple view representations as described by Roberts [1; 2] denote multiple simultaneous renderings of a data set, often using the same display technique. For example some of our view types, specifically the TreeMap and multiple tree displays, both render more than one tree using multiple instances of the same basic representation style.

Multiform visualization involves displaying the same data but through different visual representations – e.g. a tree could be shown simultaneously in different views as a TreeMap, a cone tree or as a collapsible windows explorer style tree widget.

There are numerous examples of both multiform and multiple view visualizations. Taking just the limited domain of multiple trees or hierarchies, TreeJuxtaposer [3] displays linked multiple views of a set of trees, though being all rendered in the same style do not qualify as multiform. Similarly, Sifer [4] displays a set of hierarchically facets each of which providing a different ordering on a set of shared objects. They all use the same adjacency-style display, where child nodes are placed below and abutting their parent nodes. Selections made for objects in one of the dimensions are reflected where they appear in the other dimensions and thus qualifies as an example of a linked multiple view system but again not as multiform. The Zoomology [5] browser displays two trees side by side as enclosure type visualizations along with a separate view showing them combined as a merged structure, with differences between the two trees represented as 'missing' areas, and therefore acts as both a multiple view *and* multiform visualization.

Analyzing structures that are composed of multiple smaller components can work at two levels in a multiple view environment. The first is that the separate sub-components can be visualized individually, something we have covered in previous work by viewing a collection of trees as inter-linked representations of multiple trees [6]. The second is to have multiple views of the overall structure itself, which in turn can include instances of views as described in the first approach.

It is this second approach which we describe in this paper, as we extend the idea of multiform representations for comparing trees beyond the dual tree visualization used in the Zoomology interface to several trees at once. Some of the multiform views are themselves examples of multiple views, as previously

seen in [7] where a matrix of multiple scatter-plots is included as one of the view types, but here we have trees rather than table or spreadsheet data as the fundamental structures underpinning the views.

3. Problem Domain

Taxonomists are scientists who classify organisms, and over time different taxonomists and new discoveries can lead to many different classifications arising over the same basic group of organisms. Taxonomists wishing to undertake new revisions and compare their work to old classifications must somehow deal with the complex overlaps produced between multiple classifications. Our approach is to allow taxonomists to perform dynamic queries on visualizations of the multiple classification trees

3.1. Data

The data we visualize are thus collections of overlapping taxonomic trees, where the classifications can range in size from an individual genus such as buttercups (*Ranunculus*) with roughly 300 *taxa* (singular: taxon; a node in a taxonomic tree), to annual revisions of the ITIS reference classification that aims to cover most biological species with 250,000+ *taxa*. The data as a whole forms a DAG (Directed Acyclic Graph) structure and being a collection of trees can also be termed a *Forest*.

3.2 Tasks

Taxonomists working with multiple classification trees have a number of questions they wish to answer. Some of the most common are:

T1.Which taxonomies does a taxon occur in?
T2.What are the subtaxa of a taxon in a classification?
T3.How is a taxa group dispersed in other taxonomies?
T4.What other taxa is a taxon grouped with across trees?

Any visualization that aims to help taxonomists should reveal such information clearly. In the event a single visualization cannot achieve this then perhaps multiple coordinated views can.

4. System Description

Our visualization uses a standard Model-View-Controller (MVC) approach, so each view is de-coupled from the other views, with synchronization of views taking place through the effect actions have on shared model objects and an individual view's reaction to those changes.

The basis for our Model is a Forest object, represented in Figure 1, in which each tree is a hierarchical collection of labeled nodes, with each node representing a taxon. *Forest Nodes* are inter-tree objects formed from collections of nodes with the same label that occur across several trees, and provide a convenient mechanism for communicating shared attributes of nodes, such as selection, name, taxonomic rank (genus, family etc) across trees.

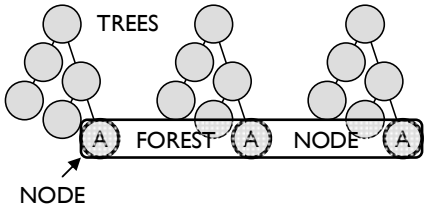

Figure 1. A Forest is composed of multiple trees. Forest Nodes span multiple trees to include a set of nodes with the same name.

4.1. Visualization Components

There are currently four views that allow exploration of all the nodes in the forest model in its entirety:

- A multiple TreeMap [8] view (Figure 2).
- A multiple adjacency tree view (Figure 3).
- A DAG view showing the forest structure (Figure 4).
- An alphabetically-ordered node list (Figure 5).

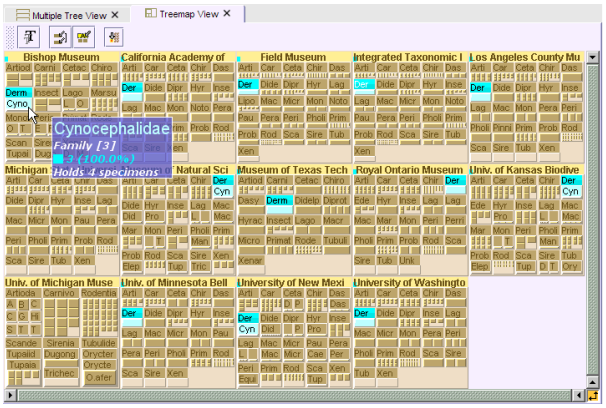

Figure 2. The TreeMap-styled view, itself a multiple view component

A fifth view, the detail panel, displays details about one node and its direct relationships, which is usually the node that was the focus of the last action. A sixth type, the history view, also queries and reacts to selections in other views but does this via another model that stores a chronological record of selections.

Of the four 'full forest' views, the TreeMap and adjacency tree views are most obviously related by

their method of displaying each tree as a separate entity, with brushing and selection on one tree reflected in the other tree representations in that view. These selections are of course also picked up by the other type of view – hence the description of the whole interface as being multiform views of multiple trees. Two range-slider widgets, adapted from the Prefuse [9] codebase, are positioned along the x and y axes to provide a simple zooming mechanism in both views.

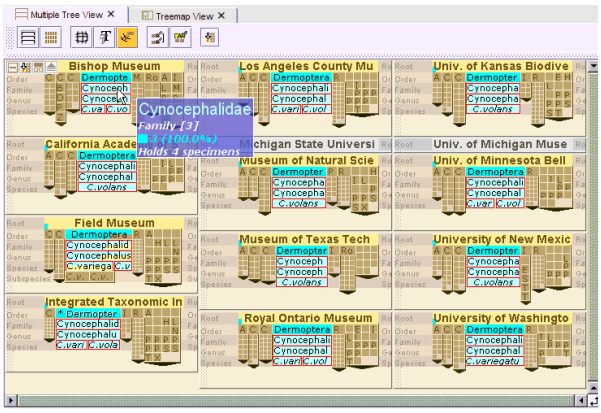

Figure 3. The adjacency tree view. Each tree has a top-down alignment compared to a TreeMap's enclosure-based layout.

The multiple TreeMap view sub-divides its area according to the number of currently visible trees. The individual TreeMaps are then drawn in their allocated areas with parent nodes having a label along the top edge of their display space, and the remainder of their area given over to recursively displaying child nodes. Figure 2 displays a multiple TreeMap view where a sub-tree in one hierarchy has been selected and coloured, with the resulting selection communicated to the other TreeMaps in the display.

Similarly, Figure 3 displays the result of the same selection propagated to a multiple adjacency tree view, showing the selection not only transmitting between trees in the same view but across to a different form of view as well. Each individual tree here is drawn top-down with child nodes placed below their parent nodes, hence the term 'adjacency' view.

The main difference between these two views is that the adjacency view displays nodes at set depths along the vertical axis according to their depth in a tree, making each representation more identifiable as a traditional hierarchy. The TreeMap view forsakes this for node density, though the difference is minimal, with the most apparent visible difference between Figures 2 and 3 being the space taken up by the depth text labels in the adjacency tree view.

The DAG and list views, shown in Figure 4 and Figure 5 respectively, appear at first glance to be completely different. However, both use the convention of amalgamating the set of trees into one visual structure rather than displaying them as distinctly individual objects as the previous views did. The trees in each view are aggregated through ForestNode objects, which span shared names common across a tree set. In the list view's case the ForestNode representations are then alphabetically organized to allow searching for a particular taxon name, while in the DAG view's case the names are organized according to the Forest's DAG structure for structural comparisons of different trees in the one view. In each case, presence and selection information is conveyed within the individual ForestNode representations

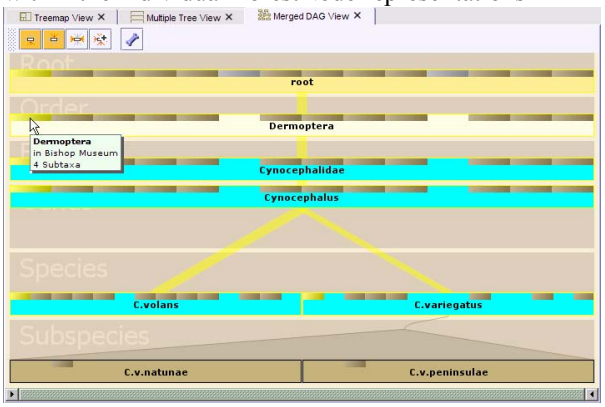

Figure 4. The DAG based view - this is a unified view of the forest structure.

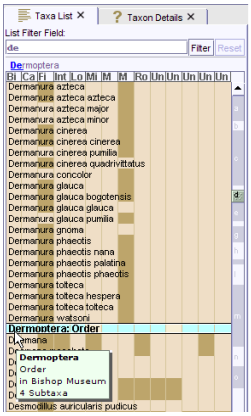

Figure 5. The list view uses shading to indicate presence of named nodes in trees.

4.2 User Interface

The user interface that can hold these visualization components initially displays with three tabbed panels, one panel on the right-hand side that is best suited for holding the list view and a pair arranged vertically on the left-hand side. The bottom one of these initially holds the history/undo view, and the top panel is

designed to reserve the most space for one of the structural view types. The relative space allocations between these panels can be adjusted with split-pane controls and the tabs rearranged within or between panels so the set-up is not fixed. This is demonstrated in Figure 6 where a DAG view has been dropped into the tab panel nominally reserved for a history view.

[10], *linked brushing* and *linked selection*. The *linked brushing* is the simplest effect, being a highlighting of temporarily selected nodes across trees and views.

Navigational slaving involves synchronizing associated views when a navigation action is performed on any one of a set of linked views. For instance, selecting node *A* in the list will re-root the

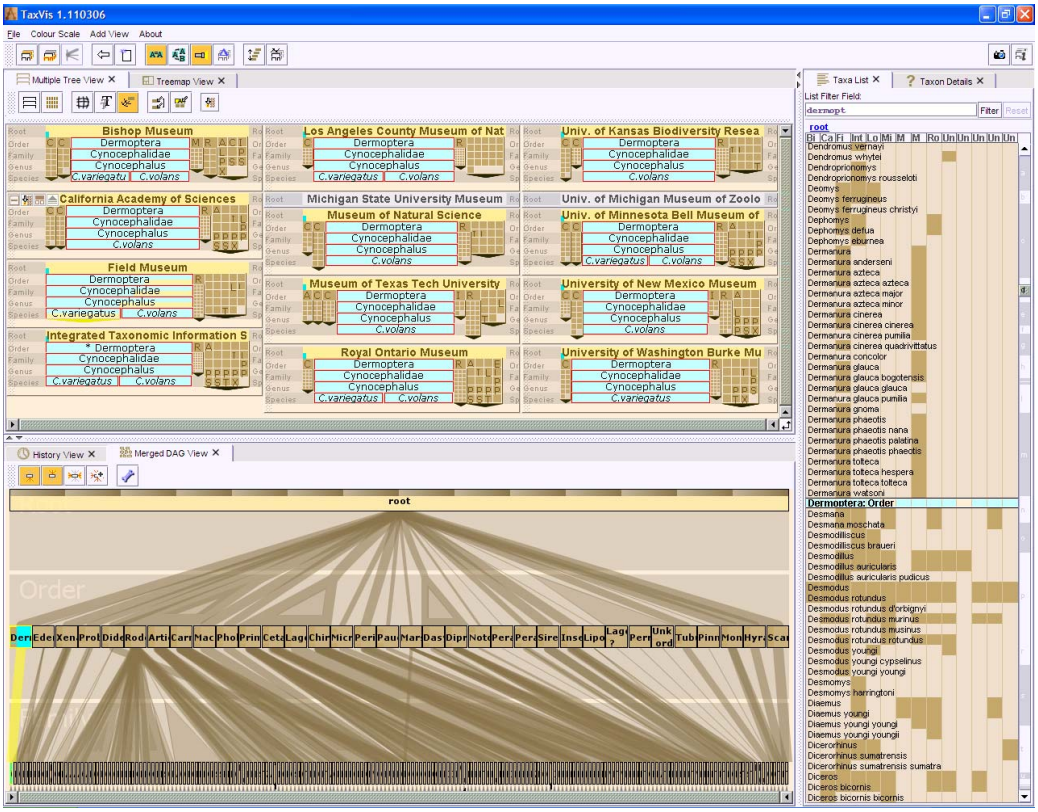

Figure 6. The TaxVis interface displaying a MANIS data set of multiple museum collections.

Above these tabbed panels is an action bar that controls functions that act on more than one view style, such as brushing, navigation synchronization within multiple tree views, and the semantics of selection operations. There are also sort order controls, which decide how nodes in the multiple tree views should be ordered (sub-tree size, name, etc) and how the trees should be sorted between themselves - chronologically or alphabetically. There are also global filters for individual tree visibility, and for the biological data which forms the bulk of our data sets, controls for filtering the display of ranks and relationship types. Finally there is a traditional menu bar for loading and saving data, and for instantiating new view instances.

4.3. Interaction

Linked interactions between the views consist of *navigational slaving*, as described in Baldonado et al

DAG view display at node *A*. In the multiple tree views in Figure 2 and Figure 3 re-rooting also applies to the individual trees displayed in the view. Each tree in the view will attempt to re-root at *A* or as closely as possible e.g. at the node that contains the contents of *A* if *A* itself is missing. The slaving does not go as far as synchronizing scroll operations. For instance, scrolling the list to nodes that begin with the letter *B* will not reveal only the same nodes in the tree-based views as the effect would be an unhelpful constant jumping of viewpoints within the views.

The same reasoning applies to *linked selections*, which can be considered a fusion of navigational slaving and linked brushing. Selecting a node in any of the views will colour it and its sub-tree permanently (until a 'clear' command) as opposed to the temporary brush selection, and will refocus each linked view to the selected node, similar to navigational slaving.

5. Discussion

Each view has strengths and weaknesses in its ability to display information about the forest structure and correspondingly will be suited to undertake a different set of tasks in comparison to the other views.

A case in point concerns the development of the list view. Over a number of think-aloud protocol tests [11] performed during this project and in previous research we have gathered feedback about various ways of displaying and interacting with multiple hierarchies. One constant aspect of our visualizations from the beginning has been the inclusion of an alphabetically-ordered list of nodes, designed to act principally as an interface element from which a user could select a starting node to then browse around the hierarchical representations we were focusing on testing.

In our tests this list often proved to be the simplest way of discovering details about the data. For instance the first task (T1) we mentioned in section 3.2 was "Which taxonomies does a taxon occur in?" Users would often answer questions in this form by just consulting the data displayed in the list; as such queries didn't require any specific probing of the tree structures. Indeed, having to interpret a display composed mainly of information redundant to a task would tend to reduce user performance, a point made in Tufte [12] about understanding data encoded in graphical designs. In this sense, for tasks that involved simply finding a named node, or finding information about a node's occurrence across the tree set, a list of node names cross-referenced by tree memberships has the advantage of showing that information and that information only.

As we realized the list was being used in this way we began to elevate the list to being a fully-fledged part of the overall visualization, allowing navigation and selection of nodes, with the difference being this navigation was performed in a linear alphabeticised space rather than in the relation-oriented space of the multiple hierarchies we were otherwise displaying.

Similarly testing of our multiple adjacency tree view and our DAG view revealed that both could perform some tasks equally well, but for others performed quite differently. As an example, one question we used during testing was "What are the members of *Selineae* in Berchtold & Presl's classification?" – based on task T2 in section 3.2 ("What are the subtaxa of a taxon in a classification?") When we asked this using the multiple adjacency view it was easy for users to find the answer as the classification was displayed separately and inside which *Selineae* would contain only its children in that classification. In the DAG view the user has to perform an extra brushing interaction to distinguish which instance of *Selineae* they are interested in as the name occurs across two trees. The difference in the views is shown in Figure 7. In both cases, users tended to make the initial selection of *Selineae* directly via the list view rather than search through the structure-based views.

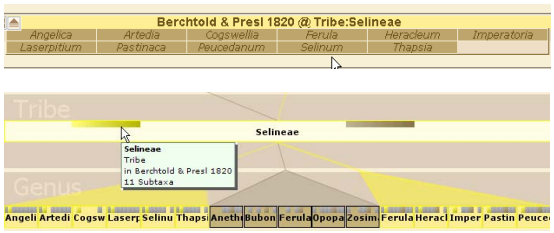

Figure 7. Comparing single tree tasks between the multiple adjacency tree and DAG views. The DAG view requires an extra interaction step.

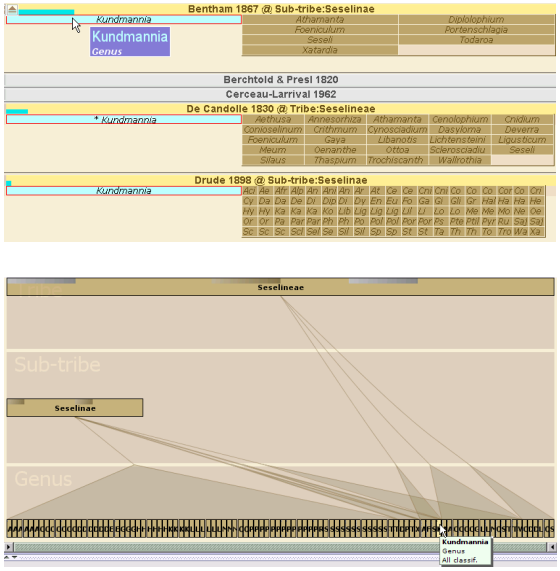

Figure 8. Comparing siblings across many trees in the adjacency tree and DAG views.

Conversely, Figure 8 demonstrates that a task derived from T4 in section 3.2 ("What other taxa is a taxon grouped with across trees?") such as "find all the siblings of *Kundmannia*" is more directly answered in the DAG view, as all the siblings of that node across multiple trees are gathered together in one place. The multiple tree representations require a user to scan the trees for the presence of *Kundmannia* and then attempt the merger of the sibling groups themselves, including finding any patterns of shared siblings etc.

The emergent pattern was that questions/tasks that involved finding information about one tree were straightforward in the multiple adjacency tree view,

whereas questions involving finding inter-tree relations were easier in the DAG view. Both representations aid certain types of tasks yet hinder others, finding answers about a single tree in the DAG view either involves visually threshing out the details of one tree from the whole structure, or activating the filter controls. Similarly, finding cross-tree information in the multiple tree views involves scanning across several distinct visual representations.

These findings further emphasize the utility of multiple, different views when exploring a complex data set, as each view will have certain tasks it allows users to perform more efficiently.

6. Conclusions

The conclusion to be reached is that there is no single "jack-of-all-trade" visualization for multiple trees that will adequately perform all the tasks a user wishes to perform, and probably no single view that can perform the tasks as efficiently as a comparable set of multiple views.

A single view of a subset of the overall information set will necessarily perform specific tasks better because it omits extraneous information that is not necessary for that task. For example, one of the multiple tree views will allow a user to see the placement of a node in the context of a single tree better than the DAG view, because the trees are separated out and shown in isolation. Looking at a single tree, the user can ignore the presence of the other trees in the forest.

Mukherjea et al [13] discussed such multiple hierarchical views were best placed to navigate and display complex structures, which we've found is true for some tasks, but they did not consider relationships formed across and between those hierarchies. This is pertinent in our work as our classifications exist as concrete objects in themselves rather than as abstractions of the larger object, thus in some cases a unified view is more appropriate.

Hence we've found that providing and combining a number of different yet complementary views is the most practical way of answering the greatest range of user queries, with the main factor involved in choosing an appropriate view simply that of understanding what is being queried. A question interrogating just the properties of a single node doesn't require structural information, and thus a list is best placed. Multiple tree views are best placed for comparing information internal to separate trees and cross-tree comparisons are best served using unified views of the forest model.

Acknowledgements

TaxVis is funded by the UK Engineering and Physical Sciences Research Council (EPSRC). Thanks to Mark Watson at RBGE for the *Apiaceae* data.

References

[1] J.C. Roberts. "Multiple-View and Multiform Visualization". In *Proc. of Visual Data Exploration and Analysis VII*, January 22-28, 2000, pp.176-185, SPIE.

[2] J.C. Roberts. "State of the Art: Coordinated & Multiple Views in Exploratory Visualization". In *Proc. of Coordinated and Multiple Views in Exploratory Visualisation*, July 2, 2007, pp.61-71, IEEE Computer Society Press.

[3] T. Munzner, F. Guimbretière, S. Tasiran, L. Zhang and Y. Zhou. "TreeJuxtaposer: Scalable Tree Comparison using Focus+Context with Guaranteed Visibility". *ACM Transactions on Graphics*, *22*(3), pp.453-462.

[4] M. Sifer. "Filter co-ordinations for exploring multi-dimensional data". *Journal of Visual Languages and Computing*, *17*(2), pp.107-125.

[5] J.Y. Hong, J. D'Andries, M. Richman and M. Westfall. "Zoomology: Comparing Two Large Hierarchical Trees". In *Proc. of IEEE InfoVis Poster Compendium*, 19-21 October, 2003, pp.120-121, IEEE Computer Society Press.

[6] M. Graham and J. Kennedy. "Combining linking & focusing techniques for a multiple hierarchy visualisation". In *Proc. of IEEE Conference on Information Visualization*, July 25-27, 2001, pp.425-432, IEEE Computer Society Press.

[7] N. Feldt, H. Pettersson, J. Johansson and M. Jern. "Tailor-made Exploratory Visualization for Statistics Sweden". In *Proc. of Coordinated and Multiple Views in Exploratory Visualisation*, July 5, 2005, pp.133-142, IEEE Computer Society Press.

[8] B. Johnson and B. Shneiderman. "Treemaps: A Space-Filling approach to the visualization of hierarchical information structures". In *Proc. of IEEE Visualization*, Oct 22-25, 1991, pp.284-291, IEEE Computer Society Press.

[9] J. Heer, S.K. Card and J.A. Landay. "prefuse: a toolkit for interactive information visualization". In *Proc. of ACM CHI*, April 2-7, 2005, pp.421-430, ACM Press.

[10] M.Q.W. Baldonado, A. Woodruff and A. Kuchinsky. "Guidelines for Using Multiple Views in Information Visualizations". In *Proc. of ACM AVI*, May 24-26, 2000, pp.110-119, ACM Press.

[11] B. Tognazzini. "User testing on the cheap". *TOG on Interface*, pp.79-89, Chapter 14. Addison-Wesley, 1992.

[12] E.R. Tufte. *The Visual Display of Quantitative Information*. Graphics Press, Cheshire, Connecticut, 1983.

[13] S. Mukherjea, J.D. Foley and S. Hudson. "Visualizing Complex Hypermedia Networks through Multiple Hierarchical Views". In *Proc. of ACM CHI*, May 7-11, 1995, pp.331-337, ACM Press.