Exploring Hierarchical Visualization Designs Using Phylogenetic Trees

Shaomeng Li[a], R. Jordan Crouser[b], Garth Griffin[b], Connor Gramazio[c], Hans-Jörg Schulz[d], Hank Childs[a], Remco Chang[b]

[a]University of Oregon, Eugene, Oregon; [b]Tufts University, Medford, Massachusetts; [c]Brown University, Providence, Rhode Island; [d]Fraunhofer IGD, Rostock, Germany

ABSTRACT

Ongoing research on information visualization has produced an ever-increasing number of visualization designs. Despite this activity, limited progress has been made in categorizing this large number of information visualizations. This makes understanding their common design features challenging, and obscures the yet unexplored areas of novel designs. With this work, we provide categorization from an evolutionary perspective, leveraging a computational model to represent evolutionary processes, the phylogenetic tree. The result — a phylogenetic tree of a design corpus of hierarchical visualizations — enables better understanding of the various design features of hierarchical information visualizations, and further illuminates the space in which the visualizations lie, through support for interactive clustering and novel design suggestions. We demonstrate these benefits with our software system, where a corpus of two-dimensional hierarchical visualization designs is constructed into a phylogenetic tree. This software system supports visual interactive clustering and suggesting for novel designs; the latter capacity is also demonstrated via collaboration with an artist who sketched new designs using our system.

Keywords: Phylogenetic Trees, Design Space Exploration, Algorithmic Design Suggestion, Visually Interactive Clustering

1. INTRODUCTION

Visualization techniques express the underlying information of a data set. As more domain scientists apply visualization techniques in different research projects, new designs keep emerging to meet the increasing variety of applications. In the *Visual Bibliography of Tree Visualization*[1] alone there are more than 275 hierarchical visualizations recorded with most having unique appearances and their own visual designs. Although the number of designs continues to grow, there has not been a very effective way for categorizing and organizing this large number of visualizations. This makes it difficult to understand their common design features, and also to explore their design space in pursuit of novel visualizations. We approach this problem by applying a tree structure to the design space connecting visualization examples in the corpus; this tree structure supports interactive visual clustering based on design features, as well as algorithmic suggestions for new visualizations in the design space.

Typically, when designing a new visualization to fit the requirements and properties of a new application scenario, visualization experts refer to existing visualizations and borrow useful design features from them. This experience is further confirmed when we examined the tree visualizations displayed in the Visual Bibliography of Tree Visualization, which contains many instances of design features inherited between different visualizations. For example, since the invention of the TreeMap[2] in 1991, many successful following visualization designs have adopted the space-filling approach from TreeMap to visualize hierarchies in two dimensions. Some notable successors include Cushion Treemaps,[3] Polar Treemaps,[4] Jigsaw Maps,[5] and Voronoi Treemaps.[6] However, beyond their similar space-filling approach, these examples also vary from each other in their own aspects: cushion visual effects help separate each node in Cushion Treemaps; Polar Treemaps adopt a circular overall layout; Jigsaw Maps produce nonrectangular regions with geometric continuity; and Voronoi Treemaps employ a Voronoi scheme to outline the regions. The existence of *inheritance* and *variance*, as shown in the above examples, is shared between visualization designs and evolutionary biology.

We propose to employ *phylogenetic analysis*, a generic technique in analyzing evolutionary processes, on visualization designs, given the properties of evolution we observed above. The key of phylogenetic analysis is a

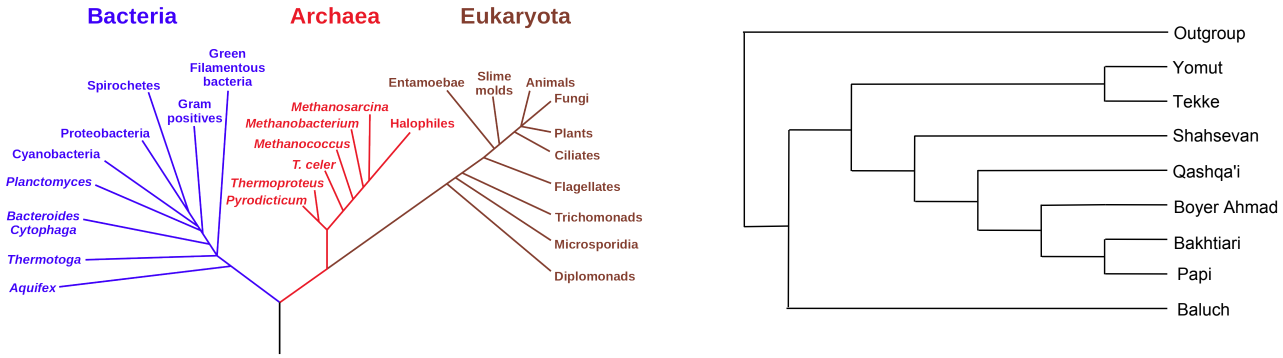


Figure 1. Examples of established phylogenetic trees. The left image, obtained from Wikipedia,[11] is a rooted *Tree of Life*.[7–9] Each leaf node represents a class of organisms. The phylogenetic tree on the right represents the evolution of woven carpet designs from different Iranian tribes.[10] It is illustrated as a cladogram to show a clearer evolution timeline: examples emerged more recently appear on the tree after a split closer to the leaf nodes.

phylogenetic tree that connects all examples with each other. It starts from the pairwise evolutionary distances between all examples, and then computes hypothetical processes of how these examples reached the current state from their ancestors through evolution. These computed processes are finally represented in a rooted or unrooted binary tree structure. A path on a phylogenetic tree from the root (rooted cases) or the center (unrooted cases) to a leaf indicates an evolutionary process from ancestors to descendants. Thus, observed examples are always at leaf positions, and internal nodes are hypothetical ancestors along the evolutionary processes. Edge lengths of a phylogenetic tree are often used to indicate the evolutionary distance from the ancestor to the descendant. Figure 1 shows two examples of established phylogenetic trees in two different applications regimes. The first one on the left, the *Tree of Life*,[7–9] provides an evolutionary history of organisms based on their rRNA genes. It shows three major clusters of organisms by the big branches (Bacteria, Archaea, and Eukaryota), and the more leaves a branch has the more diverse the evolution is in that branch. The second phylogenetic tree on the right is constructed from woven carpet designs from different Iranian tribes.[10] The carpet designs sharing a more recent common ancestor (i.e., split at a later time along the evolution) are more closely related in characteristics including decorations, materials, and fabrications. As an example, Bakhtiari and Papi are more closely related to each other than they are to Baluch according to this phylogenetic tree.

In this paper we apply phylogenetic trees to hierarchical information visualization designs. To do this, we firstly need to transform these designs into a sequence of "feature vectors," much like the rRNA genes used in the Tree of Life in Figure 1. Our approach is to identify features that describe a visualization design and then calculate the resultant feature vectors for each member in our corpus of designs. The features resulting from this process form a high-dimensional space with each member lying at a certain location. Secondly, we apply a popular phylogenetic tree construction algorithm to build the phylogenetic tree in this high-dimensional space. Thirdly, the phylogenetic tree is integrated into our software system with user interactivity and fast searching for a near-optimal tree enabled. Finally, two use cases are presented to demonstrate the usefulness of phylogenetic trees for understanding design features as well as exploring the design space in pursuing novel visualizations.

The primary contributions of this paper are two functionalities enabled by our phylogenetic analysis system: 1) algorithmic suggestions for design features of novel visualizations; and 2) support for interactive visual clustering within the design space. A secondary contribution is a systematic process for mapping visualization designs to characteristic input vectors for the phylogenetic tree construction.

2. RELATED WORK

The research in this paper is grounded in both the successful phylogenetic research on human artifacts (§2.1) and also the design space of information visualizations (§2.2).

2.1 Phylogenetic Analysis on Human Artifacts

Researchers have noticed that human artifacts evolve as human culture evolves, and widely employed phylogenetic analysis to understand this evolution. Application domains of such analysis vary from material culture in northwest Siberia[12] to popular music albums including The Beatles and other artists.[13] The research applications that most inspired our work are:

- Tehrani et al.[10] successfully clustered woven carpet designs from different Iranian tribes using a phylogenetic tree (Figure 1 right). They further discovered that the resulting phylogenetic tree shared great similarities with the linguistic relationship model among the study populations, indicating the carpet-design based phylogenetic tree was able to explain the relationships among cultural traditions.
- Riede[14] analyzed flatware production to reveal how developments in technology changed knife and fork designs from an evolutionary standpoint. They validate that both the knife and fork data sets exhibit a very strong phylogenetic signal by comparing the designs' positioning on the phylogenetic tree with the actual production order of the studied designs. The result shows they are highly comparable, which means that designs that were produced earlier are mostly positioned closer to the ancestor positions of the tree.
- Díaz-Báñez et al.[15] conducted research on Flamenco music rhythms, using phylogenetic trees to search possible missing links along the evolution of rhythms. They reconstructed the rhythm represented by one of the ancestral nodes, and eventually discovered that an almost identical rhythm had already been recorded in prior literature.

2.2 Design Space of Information Visualization

Design spaces provide a way to categorize the design choices for each visualization, thus allowing further analysis. While initial research on design spaces and new designs of information visualizations focused on a broad set of techniques,[16–19] many subsequent studies have focused on specific subsets of information visualization. Some noteworthy examples include sequential space-filling layouts,[20] glyph designs,[21] composite visualizations,[22] graphical layered elements on charts,[23] and representative factors for interactive visual analysis of high-dimensional data.[24] The research of design spaces also extends to visualization construction approaches[25] and visualization tasks,[26] whereas the motivational ones to our work are those that discovered the possibilities of new visualizations within the design spaces, notably Kerracher et al.'s research in the design space of temporal graph visualizations[27] and Schulz et al.'s generative layout approach for rooted tree drawings.[28]

The work most related to the one presented here is that of Schulz et al.,[29] who explored the design space of implicit tree visualizations. In addition to a detailed discussion of the definition and properties of this space from a more formal way, Schulz et al. also suggested the possibility of generating new techniques by combining visualization parameters and mixing design choices. However, this work differs from the one presented here as we focus on how phylogenetic trees can contribute to understanding this design space, and further lead to discovery of meaningful design choices within it.

3. CONSTRUCTING THE PHYLOGENETIC TREE

3.1 Defining Design Features

To construct a phylogenetic tree, each input to the construction algorithm — a visualization design in our case — must be described by some feature attributes with categorical values for each feature attribute. This description is natural in some applications, DNA sequences in evolutionary biology for example, but there is no natural description for visualization designs. Luckily, the design space by Schulz et al.[29] serves as a good starting point for defining our feature attributes. From that design space, we eventually borrowed two feature attributes (*Hierarchy Representation* and *Primitive Shape*) into our set of features.

To better define a visualization, we added more feature attributes based on our knowledge and experience. The attributes we focused on were mostly semantic descriptions of visual aspects of the visualization designs, like the *Contour Regularity* of a visualization. We did not incorporate feature attributes that concern the process for

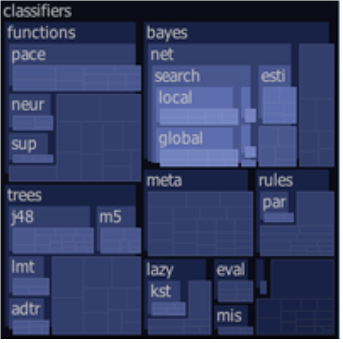
Feature Attributes	Binary Feature Values	Feature Vector for Cascaded Treemap	
Contour Regularity	*Free* or *Regular*	*Regular*	 classifiers functions bayes pace net search esti local neur global sup meta rules trees par j48 m5 lmt lazy eval kst adtr mis
Hierarchy Representation	*Adjacency* or *Inclusion*	*Inclusion*	
Primitive Shape	*Triangle* or *Non-Triangle*	*Non-Triangle*	
Primitive Shape	*Rectangle* or *Non-Rectangle*	*Rectangle*	
Primitive Shape	*Circle* or *Non-Circle*	*Non-Circle*	
Primitive Shape	*Arc* or *Non-Arc*	*Non-Arc*	
Layout Direction	*Radial* or *Not*	*Non-Radial*	
Layout Direction	*Axis-Parallel* or *Not*	*Axis-Parallel*	
Node Packing	*Tightly* or *Loosely*	*Loosely*	
Strict Symmetrical	*Is* or *Is Not*	*Is Not*	
Leaves Encoding Detailed Info	*Is* or *Is Not*	*Is Not*	
Special Visual Effects	*Has* or *Has Not*	*Has (Transparency)*	

Figure 2. Features chosen for our two-dimensional implicit tree visualizations. The left column shows the specific feature attributes; the middle column shows the two possible values for each feature attribute; the right column shows the feature values for the Cascaded Treemap,[30] a member of our corpus of visualization. The collection of feature values, such as ones in the right column, is referred to as a feature vector. Feature attributes with more than two values (e.g., *Primitive Shape*) are flattened into multiple ones so that feature values for each attribute could be binary.

generating the visualization, the types of data presented, etc.; our emphasis was just on what could be seen. We also chose visual aspects with a reasonable amount of generality. That is to say, a visual aspect only applicable to a single example was not included. Finally, we excluded *Color* from our feature set, since we believe color to be a research field in its own right that 1) differs from the design of a visualization we focus here; and 2) cannot be captured with a fairly good quality by only a few categorical values.

Each design feature attribute contains categorical values indicating the multiple design choices for that attribute. Ideally, the features used to describe a visualization are independent, meaning whatever value a visualization design takes from one feature attribute does not create a constraint for other attributes. However, this requirement creates a tension. On the one hand, a larger number of feature attributes is desirable for a feature set, because more attributes increase the ability to describe a visualization. On the other hand, the fewer the feature attributes, the easier it is to maintain their independence, since each additional attribute must be independent from all previous selections. We note that we could not fully achieve independence, since some feature attribute combinations always require a certain value for another feature attribute, but not vice versa. Our decision to not pursue fully independent features was partially due to successes in other research projects where non-independent features were used in computational tasks. Evolutionary study of flatware designs[14] and user re-authentication via mouse movement profiles[31] are such examples.

Each of the resulting design feature attributes has categorical values — some are binary and others have more than two values. When it comes to representing these categorical values numerically for tree construction purpose, it is intuitive to map each value to an integer, like feature values $\{Free, Regular\}$ of the attribute *Contour Regularity* could be mapped as $\{0, 1\}$ respectively. In this binary case we easily define that *Free Contour* is 1 distance away from *Regular Contour*, and vice versa if there is a feature value of distance 1 from *Regular Contour* it must be *Free Contour*. However, in the case of feature attributes having multiple values we lose both above abilities of straightforward distance definition and unambiguous feature value referring. This disparity makes it hard to perform computation for constructing the phylogenetic tree, and also introduces uncertainty when it comes to suggest design feature values algorithmically (see §4.3), since both procedures require precise knowledge of the distance between two examples.

To address this more-than-two-value problem, we adopt a common technique from machine learning to create uniform features with only binary values: for each feature attribute with k possible values, we flatten this attribute into k binary ones with each representing the *presence* or *absence* of that particular feature attribute. In phylogenetic analysis, the work by Tehrani et al.[10] on evolution of Iranian tribal carpets is an example of successful application of this technique. Another benefit from this technique is that we can capture more design possibilities: a visualization that utilizes both *Rectangles* and *Circles* cannot be accurately depicted with only one *Primitive Shape* feature attribute that contains all shapes.

After defining and adjusting features as discussed above, our feature set consists of 12 binary features. The resulting feature attributes, their binary values, and the feature vector of a sample visualization are shown in Figure 2. Of course, the features we describe here are simply an attempt to yield a numerical descriptor for the graphical design of tree visualizations. Their biggest value is in providing a concrete example on how phylogenetic trees can be used to facilitate our understanding of the design features and exploration of novel designs. A broader study that includes other categories of visualizations would likely require additional considerations of design features; that said, the design feature set we defined here would likely be a good starting point.

3.2 Selecting an Example Visualization Corpus

Selecting an example visualization corpus is a two-step process: first, identifying the visualizations to use for our study and, second, refining duplicate examples by their feature vectors.

Identifying Visualizations: When choosing the visualization instances to form our corpus, our goal is that the corpus size is appropriate, meaning that: 1) it is large enough to provide sufficient diversity; and 2) it is small enough for us to closely study instances with commonalities. We chose the group of two-dimensional implicit hierarchical visualizations for our study since it met our corpus size goal. To clarify, implicit hierarchical visualizations mean that they use relative positioning rather than explicit edges to encode the parent-child relationship. As an example, the Cascaded Treemap[30] in Figure 2 positions individual instances inside of the categories to which they belong, indicating the hierarchy of the structure. We chose to include only visualizations that have emerged since the early 1980s, when graphics and visualization research began a period of significant innovation. We started our visualization identification from the Visual Bibliography of Tree Visualization, which provides a collection of such works.

Since our focus was strictly on innovation of design features for visualizations, we eliminated instances that fell into the following criteria: 1) examples that are mimics of natural trees; 2) examples that are essentially user interfaces; and 3) examples that do not contain novelty in design features (e.g., their novelty was in algorithmic and interactive improvements). Respective example visualizations eliminated by these three criteria include Botanical Tree Layout,[32] PygmyBrowser,[33] and Zoomable Treemap.[34] Further, another four visualizations were excluded for individual concerns, including TreemapBar,[35] since it was not strictly a hierarchical visualization. Overall 28 examples were eliminated by this process.

Refining Duplications: Each visualization design in our identified corpus has a value for each feature attribute defined in §3.1, creating a binary feature vector for this visualization. Unfortunately, some visualizations shared the same feature vector. This did not mean that the two visualizations were identical; rather, there is a difference that our feature attributes are not able to capture, because of the constraints discussed in §3.1. It is worth noting that this identical-feature-vector problem also provides clues for us to refine our chosen design feature attributes. Within the constraints discussed in §3.1, we tweaked the design feature set to eliminate as few examples as possible. Finally, when we encountered such duplications, we retained only the chronologically earliest visualization to represent that feature vector. Examples eliminated in this phase included Ellimaps[36] from 2007, as its feature vector was identical to Treemap with Ovals[4] from 1993. After removing another 23 duplications, we eventually had 35 example visualizations remaining in our corpus.

3.3 Phylogenetic Tree Construction

With a feature set defined and an example visualization corpus to study, we had the necessary ingredients to reconstruct the phylogeny. Here we used the popular Neighbor-Joining (NJ)[38] algorithm to construct a phylogenetic tree. The NJ algorithm takes as input the predefined evolutionary distance between each pair of instances in our corpus. For our application, we leveraged the binary feature values and defined the evolutionary distance between two examples to be the Hamming distance[39] between their feature vectors. Using this input, the NJ algorithm iteratively joins pairs of examples using a process similar to that of agglomerative hierarchical clustering, resulting in a tree with all examples on the leaf positions. Figure 3 provides an example of iteratively constructing a phylogenetic tree using the NJ algorithm.

The resulting phylogenetic tree has meaningful edge lengths, i.e., the edge length between two nodes represents the theoretical evolutionary distance between them. That being said, by adjusting the edge lengths at every iteration of the tree construction phase, the length of the path connecting two examples on leaf positions of

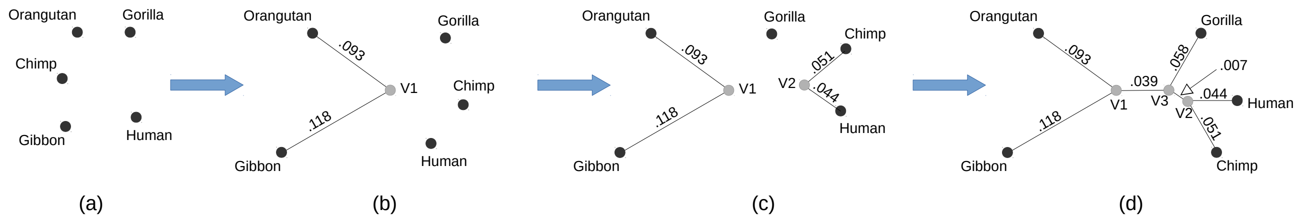


Figure 3. Illustration of iteratively constructing a phylogenetic tree using the NJ algorithm. Distances between mitochondrial DNA sequences of five primate species[37] are used in this example. The NJ algorithm starts from where all five species are separate, as shown in (a). It then selects two nodes based on current pairwise distances to connect to a newly created parent node with calculated edge lengths, as shown in (b). The parent node now represents itself and its two children to calculate distance with every other node in this corpus. This process iterates until there are only three nodes left ($V1$, $V2$, and *Gorilla*), as shown in (c). The final step is to create a last node to connect the remaining three nodes together to finish the tree construction. $V3$ is the last created node in (d). The resulting tree has the nodes representing five primate species as leaves, while newly created parent nodes ($V1$, $V2$, and $V3$) possess internal positions.

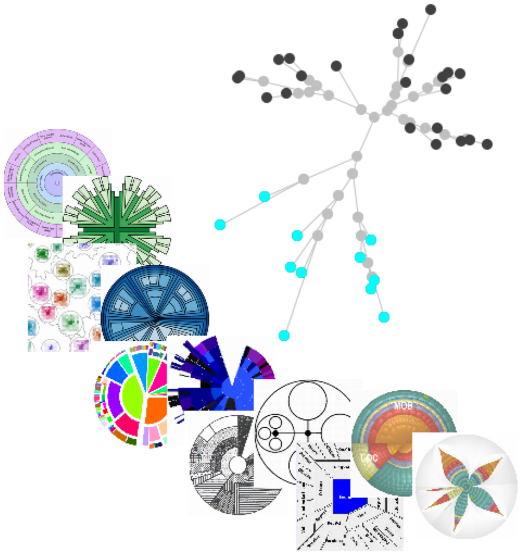


Figure 4. An example of the constructed phylogenetic tree and thumbnails representing the selected leaf nodes (in cyan). This tree is laid out adopting a radial layout,[43] with edge lengths encoding the theoretical evolutionary distance. The gray nodes are internal nodes generated by the algorithm, while black and cyan nodes are leaves representing visualization examples from our corpus. Leaf nodes become cyan when under selection, with the corresponding thumbnails showing next to them. In this screenshot, all leaves in the lower branch are selected. Thumbnails further indicate that this is a branch of visualizations featuring a *Radial Layout Direction.*

the tree is also preserved as close to their evolutionary distance as possible. This property was essential for the visual clustering support discussed later in §5.1. Finally, we notice that the NJ algorithm has a computational complexity of $O(n^3)$ for an example corpus of size n. Our implementation manages to construct a phylogenetic tree instantaneously due to a moderate data size. However, applications in a larger scale might consider one of the heuristic variances of the NJ algorithm that have lower computational complexity.[40–42]

We render the constructed phylogenetic structure adopting a radial layout[43] as shown in Figure 4, where the tree appears as unrooted with branches emanating toward the outside from its center. Using this layout, the internal nodes are the hypothetical ancestors (in gray), and a path from the center to a leaf represents a hypothetical evolutionary process. Leaves are in black by default, and turn into cyan when selected. Note that some nodes are overlapping because of the non-uniform distribution of edge lengths. This drawback can be overcome by zooming-in on a particular area of interest.

4. A SOFTWARE SYSTEM FOR PHYLOGENETIC ANALYSIS

4.1 System Overview for Phylogenetic Analysis

Our software system builds the phylogenetic tree and enables users to interact with it. Figure 5 shows a screenshot of its interface. The layout and interactive features of this system are described further in this section. We believe its noteworthy features are its visual clustering capabilities and the suggestions it can provide for inspiring new

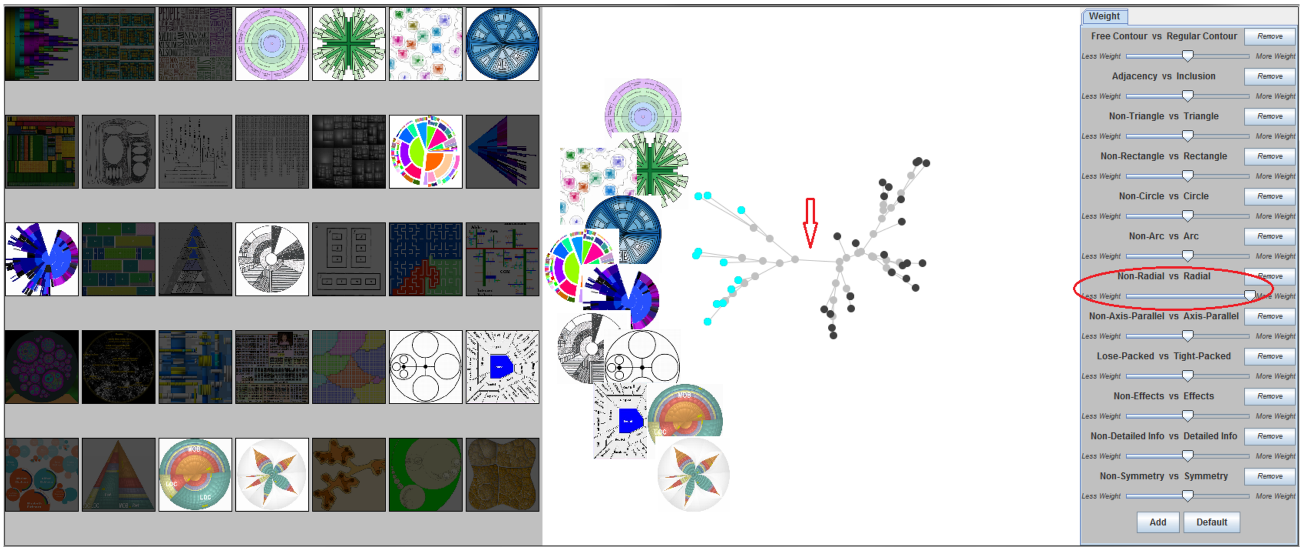


Figure 5. A screenshot of our phylogenetic analysis system, which examines features on a set of visualization designs. In this example, the user has given dominant weight to the feature *Layout Direction: Radial or Not-Radial* using the sliding bar (as circled in red). This causes the updated phylogenetic tree in the central panel to split into two partitions based on values for this feature. The separation can be clearly seen, because the edge between the two clusters is relatively long. (We indicate this edge in the image above with a red arrow.) By selecting one of the partitions (in cyan) to see the visualizations they represent, the user is able to identify that the partition on the left features *Radial Layout Direction*. This highlighted selection is also shown in the overview panel on the left.

visualizations; both are described in §5. Our system uses three panels to display an overview of corpus, a phylogenetic tree, and the feature information:

- The left panel provides an overview of examples in the corpus by showing their thumbnails. When one or more examples are selected either from this left panel or central panel where the phylogenetic tree is displayed, the unselected thumbnails are de-emphasized by graying them out to highlight the selected ones.
- The right panel enables adjusting weights of each single design feature in the feature set, which eventually results a reconstructed phylogenetic tree to be shown in the central panel. When a single node of the tree is selected, depending on whether the node is a leaf or an internal node, this right panel provides the detailed feature values of the selected example (leaf case) or suggested feature values for the hypothetical ancestor (internal node case).
- The central panel displays the phylogenetic tree that bridges the left and right panel. Zooming is supported to facilitate examining of the tree at different levels of detail. Thumbnails represented by the selected nodes are put next to the node selection. The main purpose of the central panel is to create visual representations that can aid mental models of an organized visualization corpus, as well as the underlying design space.

Our software system augments its interactive capabilities by allowing users to adjust the importance of each design feature based on individual needs. By default, construction of the phylogenetic tree assumes that all features are equally important. However, in our system, experts can steer the construction process by informing the algorithm that some features are more important than others. Recall that the pairwise distance of two examples in the tree construction step (§3.3) is determined by the Hamming distance of their feature vectors; the different feature importance in our system is achieved by imposing a weight to each feature when calculating the Hamming distance. As an example shown in Figure 5, the user assigns the feature *Layout Direction: Radial or Non-Radial* a much higher weight than the others, and successfully splits the tree into two major parts separated by a relatively long edge (indicated by a red arrow). Thumbnails show that the left split contains all *Radial Layout Direction* examples. This control also enables the capabilities described in §4.2. Finally, the user is also able to add additional features or remove existing ones through the right panel.

4.2 Fast Searching for Optimal Feature Weights

Although our software system allows users to interactively adjust design feature weights based on their individual need and experience, it also can provide a good initial weighting. The tool can search through many possible weight combinations, and correlate the resulting tree layouts with the publication year of each visualization example in our corpus. The tool selects the weight combination that creates an evolutionary history that most closely matches the chronological order.

The number of possible choices for weight combinations is very large (i.e., exponential to the number of design features in our feature set), so the tool uses a statistical sampling technique: *Latin Hypercube Sampling (LHS)*,[44] which provides good coverage and efficient sampling of such spaces. With LHS, each feature weight is discretized in a certain range and LHS constructs combinations of weights as part of the input to the initial tree construction process. After exploring the outputs of LHS, we found that 100 intervals — and thus 100 weight combinations, based on how LHS works — was sufficient to find trees that scored well based on our measurement. This LHS sampling process took approximately two seconds, which, as a one-time operation, we felt was acceptable for interactive use.

This additional complexity — incorporating weights, LHS, and comparing trees to chronological history — benefits users primarily by creating an initial phylogenetic tree that better reflects chronological history, which we believe is a better starting point. Figures 4 and 5 demonstrate LHS in practice. In Figure 5, we increase the weight for *Layout Direction: Radial or Non-Radial*, but keep all the other weights equal by default. Figure 4 also increases the weight of the same feature, but we run LHS on the remaining weights. The result is that Figure 4 has the same clear clustering of *Radial Layout Direction* visualizations as the phylogenetic tree in Figure 5, but it does so with a tree that better reflects the chronology.

4.3 Algorithmic Suggestion for Novel Visualization Designs

While leaf nodes of a phylogenetic tree are from our visualization corpus and their feature vectors are defined before the tree construction, the internal nodes of the tree are automatically calculated and placed on the phylogeny as ancestors of the leaves. They provide us with clear evolutionary paths along the tree, but do not come with the feature vector information automatically. We believe these internal nodes are of extra importance in designing novel visualizations, and thus included in our system a means to compute possible feature vectors for these internal nodes as design suggestions to the user.

To estimate the feature vector for an internal node, we employed a brute-force algorithm to filter out the feature vector that best matches its evolutionary position in the phylogenetic tree. For a d-dimensional binary feature space, there are 2^d unique feature vectors. The algorithm tries every possible feature vector and ranks them based on the discrepancy between the distances with all leaves in the design space (which are essentially Hamming distances of their feature vectors) and the distances on the phylogenetic tree (which are the path lengths to every other leaf). The feature vector that ranks first is reported on the right panel to the user as a novel design suggestion.

Although this calculation is expensive, it is only triggered when one single internal node is selected, and only computes the feature vector for that particular internal node. This calculation can be finished within one second, which we deem acceptable for interactive use since the user usually requires significantly more time examining the resulting design suggestion.

5. USE CASES FOR EXPLORING VISUALIZATION DESIGNS WITH A PHYLOGENETIC TREE

We present two use cases that motivate the usefulness of phylogenetic trees for understanding and exploring visualization designs of our example corpus: using phylogenetic trees for interactive visual clustering (§5.1) and novel visualization discovery (§5.2).

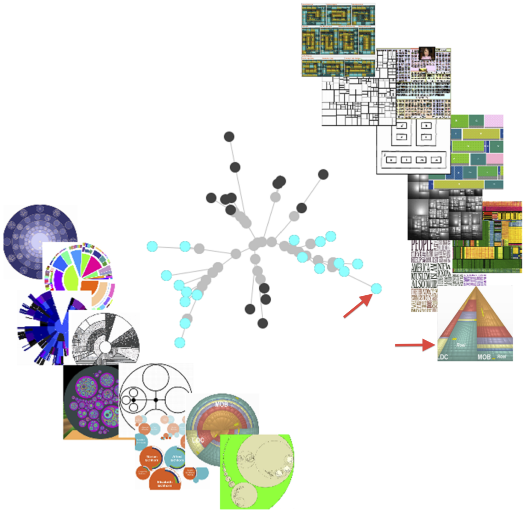


Figure 6. Two example clusters given by our phylogenetic tree. Leaves of the selected clusters are rendered in cyan, and thumbnails of these leaves are displayed around them. It shows that the left cluster consists of round-looking visualizations, while the right cluster contains rectangular-looking ones. Note the thumbnail of Generalized Treemap (Pyramid)[45] indicated by a red arrow is represented by the rightmost node on the tree, which is also pointed by a red arrow. It is essentially created by generalizing a regular "Squarified Treemap"[46] into a triangle outline, but still keeps every other property of a Squarified Treemap. This triangle appearance explains why it lies further from other examples in its cluster.

5.1 Using Phylogenetic Trees for Interactive Visual Clustering

Phylogenetic trees can be used to help a user better perceive clustering results by visually positioning closely related nodes together. The additional capability to make this improvement comes from the NJ algorithm that builds the phylogenetic tree (see §3.3). As previously discussed, the NJ algorithm takes evolutionary distances (Hamming distance of two feature vectors in our case) for every pair of examples as input. It uses these distances for several purposes, but one of them is to assign certain length values to every edge in the resulting tree. These lengths measure the closeness of two nodes: small lengths denote close pairs of nodes, while large lengths denote distant pairs. As a result, when drawing the phylogenetic tree on a flat plane, nodes are rendered at a proportional distance to the closeness of their pairwise distance. This positioning property of phylogenetic trees has been successfully used in clustering tasks, including the work by Cuadros et al.,[47] where a large amount of text documents are projected on a two-dimensional plane and clustered into meaningful categories.

Figure 6 shows an example of the visual clustering enabled by phylogenetic trees. Two distinct clusters of nodes are selected, both shown in cyan. With this view, the left branch consists of round-looking visualizations, while the right branch contains examples that are rectangular-looking. One advantage of encoding ancestral closeness of edge lengths is that users can immediately grasp their relationship, by seeing which nodes in a cluster are tightly packed and which are more loosely coupled. In the example shown in Figure 6, the rightmost selected node (indicated by a red arrow) lies relatively far away from the others in its cluster. Further examination shows that this visualization, Generalized Treemap (Pyramid)[45] which is the thumbnail indicated by a red arrow, is actually a "Squarified Treemap"[46] generalized into a triangle outline but still keeps every other property of a Squarified Treemap. This is why the node was clustered with other rectangular visualizations, and also explains its relatively long distance to other examples in its cluster. With the ability to interactively adjust the design feature set both in terms of number of features as well as the weighted contribution of each feature (§4.1), the user is able to interactively customize and examine the clustering result based on individual needs.

5.2 Using Phylogenetic Trees for Novel Visualization Discovery

By selecting a set of features to depict the visualization designs in §3.1, we essentially define a high-dimensional design space, with each feature being one dimension of this design space. The design space is quite sparse though, with only dozens of instances out of thousands of possibilities. Motivated by the sparseness of this design space, we studied how the phylogenetic tree can be used to explore the space and further provide suggestions for novel visualization designs.

The construction process for phylogenetic trees creates hypothetical internal nodes that serve as ancestors to the leaf nodes. The intuition is that, as an evolutionary process, the leaf nodes — the example visualization designs from our corpus — are descendants of some internal nodes that were not observed — ancestral visualizations that share common features. From the perspective of the design space, internal nodes are important

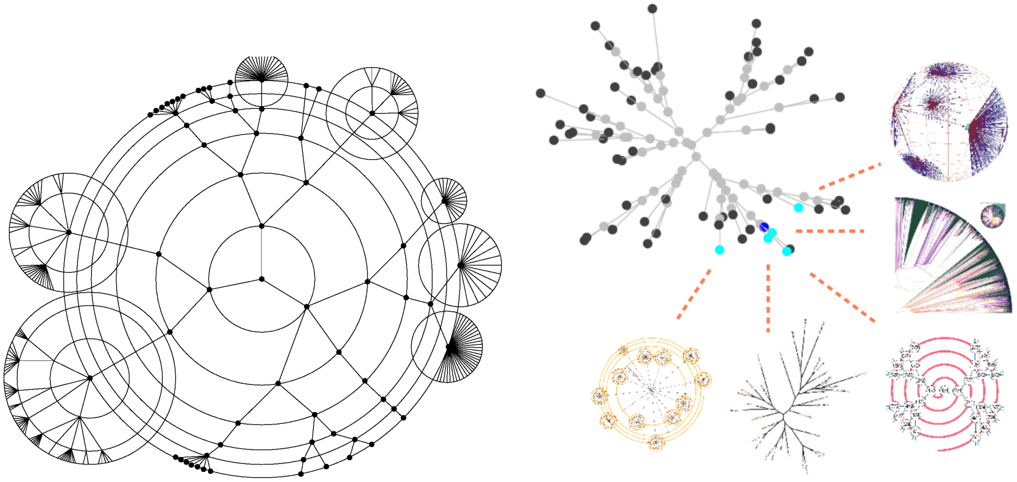


Figure 7. An artist's sketch of a novel visualization for a tree exhibiting high branching factor near its leaves (left). A screenshot of the phylogenetic tree that the artist used to explore the design space is displayed on the right. The five thumbnail visualizations are examples that have contributed to the artist's novel design. Their positions in the phylogenetic tree are indicated in cyan.

players in the evolutionary process and are likely to represent valid visualizations, as indicated by the viability of their descendants. The motivating idea here is to use our system for exploring phylogenetic trees to isolate the common descendants and display them, possibly stimulating new ideas for visualizations based on their commonalities. Such speculative work has provided outstanding dividends in other areas: the *fandango de Huelva* rhythm was (re-)discovered by reconstructing an ancestral rhythm from a phylogenetic tree of the Flamenco class of rhythm.[15]

With the functionality of algorithmic suggestion for novel designs implemented in our software system (§4.3), we present two sample usage scenarios to demonstrate the effectiveness of this technique. The scenarios each involve enlisting an artist's help in designing new visualizations to handle hypothetical data sets exhibiting particular characteristics. The artist used our tool as part of his design process and recorded how he approached the tasks. For each of the two scenarios, we present the initial requirements, the portion of the design space that influenced the artist's design, and the resulting sketch by the artist. Finally, we note that the visualization corpus in these two scenarios is different from the one presented in §3.2, specifically in that it contains more examples. This is because our feature choices varied while we conducted research on this topic, which results in changes of the example corpus. However, the artist who collaborated with us at an earlier stage is no longer available. We feel this discrepancy does not weaken the argument presented here, since the results would ultimately be similar.

Scenario 1: Visualization design for trees with high branching factor near leaf nodes. The first design scenario is to design a tree visualization for a hypothetical hierarchical data set with the property that the branching factor of the tree increases rapidly near the lower hierarchy levels of the tree. That is, the number of children becomes very large when internal nodes get closer to the leaves. This kind of hierarchies appear often in managerial structures. The artist used our software system to explore existing relevant visualizations and designed a new visualization for this hypothetical data set. The artist reported that his design was based on five existing visualizations because of their matching prominent feature. Of equal importance he consulted the design features for some internal nodes of this branch and finally decided to take one of the suggestions. The design and the five visualizations on which it was based are shown in Figure 7.

Scenario 2: Visualization for deep trees with low branching factor.

The second design scenario is to design a tree visualization for a hypothetical hierarchical data set with a low branching factor but very large depth. Again, the artist used our system to design a new visualization for this hypothetical data set. The artist again reported that his design was inspired by three existing visualizations and

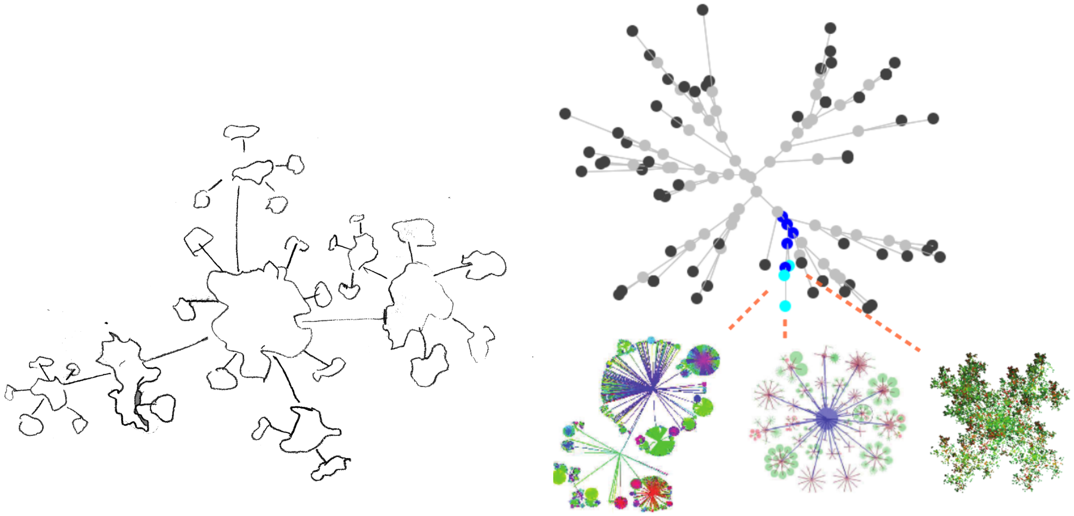


Figure 8. An artist's sketch of a novel visualization for a hierarchy with a low branching factor and large depth (left). A screenshot of the phylogenetic tree that the artist used to explore the design space is displayed on the right. The three thumbnail visualizations are examples that have contributed to the artist's novel design. Their positions in the phylogenetic tree are indicated in cyan.

suggested feature values of one of their ancestors. The design and the three visualizations on which was based are shown in Figure 8.

Artistic Design Process. While performing each of the above tasks, the artist took notes on aspects of the design process, and subsequently described the experience to us. As the artist's execution followed a similar pattern for both tasks, we do not describe the individual scenarios but instead present a summary of the artist's process for both tasks:

1. The artist typically began examining some existing visualizations by scanning the overview panel and their corresponding leaf nodes in the phylogenetic tree. The artist reported that this was to get a sense of what approaches had already been used to address similar problems to the one posed by his task.

2. Once the artist found a few examples that he deemed relevant, the next step was to examine their common design features and to increase their features' weights to generate new phylogenetic trees. The artist indicated that this gave a clearer cluster that helped gain a better sense of what are important design features and how other visualizations utilize these features.

3. The artist then tweaked the weights to exclude some irrelevant examples from the current branch, while, at the same time, examining values of feature vectors reported for some internal nodes on this branch. The artist reported this process as looking for the right "parent" of the leaf nodes.

4. After iterating on steps two and three for several times, the artist copied down the feature values reported for a selected internal node and then spent some time writing verbal descriptions of what the feature values implied in terms of the visualization design.

5. The artist then made some rough sketches, which then underwent one or two revisions to arrive at the final sketch.

A large portion of the time the artist spent using the tool was in exploring nodes and tweaking weights of each of the design features. The artist reported that the exploration provided a good way to understand the clusters implied by the subtrees and being able to tweak the weights further facilitated this process. While exploring the tree during the second usage scenario, the artist reported recognizing that some structures were in an area near where they were in the first scenario. This shows that the artist was building a mental model of the design space.

The design process also demonstrated substantial use of the internal nodes generated by the system. In particular, step three shows that the artist conceptualized internal nodes as joining multiple subtrees. Further, the artist's repeated use of the feature vectors reported by the tool for internal nodes is evidence that having quantitative information about potential novel visualizations is helpful for designing novel visualizations.

6. CONCLUSIONS AND FUTURE WORK

This work explores the phylogenetic tree as a computational model for understanding information visualization designs. Within our visualization example corpus — two-dimensional hierarchical visualizations — the phylogenetic tree demonstrates its usefulness in understanding visualization designs, specifically for enabling user-interactivity for visual clustering analysis, and for suggesting instance feature values that might lead to new visualizations. This paper also contributes in describing how to apply phylogenetic analysis to visualization designs with considerations in mapping visualization designs onto a valid numeric input for the phylogenetic tree construction. A final contribution of this paper is a description of a software system that enhances visual exploration and analysis of phylogenetic trees.

In terms of future work, we would like to better understand the evolutionary order of the visualization designs implied by the phylogenetic tree. This requires incorporating more information about the visualization designs into the analysis, including, but not limited to, the publication date, main contributions, and which designs influenced each other. Further, new methods to quantify these factors would also be required. Finally, we limited our analysis to a small subset of visualizations, and we believe studying a larger set of visualizations could lead to even more interesting results.

ACKNOWLEDGMENTS

Work on this research has been funded in part by the federal state of Mecklenburg-Vorpommern and EFRE within the project "Basic and Applied Research in Interactive Document Engineering and Maritime Graphics" and by the German Research Foundation (DFG).

REFERENCES

[1] Schulz, H.-J., "Treevis.net: A tree visualization reference," *IEEE Computer Graphics & Applications* **31**(6), 11–15 (2011).

[2] Johnson, B. and Shneiderman, B., "Tree-maps: A space-filling approach to the visualization of hierarchical information structures," in [*Visualization, 1991. Visualization'91, Proceedings., IEEE Conference on*], 284–291, IEEE (1991).

[3] van Wijk, J. J. and van de Wetering, H., "Cushion treemaps: Visualization of hierarchical information," in [*Information Visualization, 1999.(Info Vis' 99) Proceedings. 1999 IEEE Symposium on*], 73–78, IEEE (1999).

[4] Johnson, B. S., *Treemaps: visualizing hierarchical and categorical data*, PhD thesis, University of Maryland at College Park (1993).

[5] Wattenberg, M., "A note on space-filling visualizations and space-filling curves," in [*Information Visualization, 2005. INFOVIS 2005. IEEE Symposium on*], 181–186, IEEE (2005).

[6] Balzer, M., Deussen, O., and Lewerentz, C., "Voronoi treemaps for the visualization of software metrics," in [*Proceedings of the Symposium on Software visualization*], 165–172, ACM (2005).

[7] Lecointre, G. and Le Guyader, H., [*The tree of life: a phylogenetic classification*], vol. 20, Harvard University Press (2006).

[8] Doolittle, W. F. and Bapteste, E., "Pattern pluralism and the tree of life hypothesis," *Proceedings of the National Academy of Sciences* **104**(7), 2043–2049 (2007).

[9] Delsuc, F., Brinkmann, H., and Philippe, H., "Phylogenomics and the reconstruction of the tree of life," *Nature Reviews Genetics* **6**(5), 361–375 (2005).

[10] Tehrani, J. and Collard, M., "The evolution of material culture diversity among Iranian tribal populations," in [*Pattern and Process in Cultural Evolution*], 99–112, University of California Press (2009).

[11] Wikipedia, "Tree of life — Wikipedia, the free encyclopedia," (2004). [Online; accessed 18-July-2014].

[12] Jordan, P., "Linking pattern to process in cultural evolution: explaining material culture diversity among the northern khanty of northwest siberia," in [*Pattern and Process in Cultural Evolution*], 61–84, University of California Press (2009).

[13] George, J. and Shamir, L., "Computer analysis of similarities between albums in popular music," *Pattern Recognition Letters* **45**, 78–84 (2014).

[14] Riede, F., "Tangled trees: Modelling material culture evolution as host-associate co-speciation," in [*Pattern and Process in Cultural Evolution*], 85–98, University of California Press (2009).

[15] Díaz-Báñez, M., Farigu, G., Gómez, F., Rappaport, D., and Toussaint, G. T., "El compás flamenco: a phylogenetic analysis," in [*Proceedings of BRIDGES: Mathematical Connections in Art, Music and Science*], 61–70 (2004).

[16] Card, S. K. and Mackinlay, J., "The structure of the information visualization design space," in [*Information Visualization, 1997. Proceedings., IEEE Symposium on*], 92–99, IEEE (1997).

[17] Chi, E. H.-h., Barry, P., Riedl, J., and Konstan, J., "A spreadsheet approach to information visualization," in [*Information Visualization, 1997. Proceedings., IEEE Symposium on*], 17–24, IEEE (1997).

[18] Chi, E. H.-h., "A taxonomy of visualization techniques using the data state reference model," in [*Information Visualization, 2000. InfoVis 2000. IEEE Symposium on*], 69–75, IEEE (2000).

[19] Fujishiro, I., Ichikawa, Y., Furuhata, R., and Takeshima, Y., "Gadget/IV: A taxonomic approach to semi-automatic design of information visualization applications using modular visualization environment," in [*Information Visualization, 2000. InfoVis 2000. IEEE Symposium on*], 77–83, IEEE (2000).

[20] Baudel, T. and Broeksema, B., "Capturing the design space of sequential space-filling layouts," *IEEE Transactions on Visualization and Computer Graphics* **18**(12), 2593–2602 (2012).

[21] Maguire, E., Rocca-Serra, P., Sansone, S., Davies, J., and Chen, M., "Taxonomy-based glyph design – with a case study on visualizing workflows of biological experiments," *IEEE Transactions on Visualization and Computer Graphics* **18**(12), 2603–2612 (2012).

[22] Javed, W. and Elmqvist, N., "Exploring the design space of composite visualization," in [*Pacific Visualization Symposium (PacificVis), 2012 IEEE*], 1–8, IEEE (2012).

[23] Kong, N. and Agrawala, M., "Graphical Overlays: Using Layered Elements to Aid Chart Reading," *IEEE Transactions on Visualization and Computer Graphics* **18**(12), 2631–2638 (2012).

[24] Turkay, C., Lundervold, A., Lundervold, A., and Hauser, H., "Representative factor generation for the interactive visual analysis of high-dimensional data," *IEEE Transactions on Visualization and Computer Graphics* **18**(12), 2621–2630 (2012).

[25] Grammel, L., Bennett, C., Tory, M., and Storey, M.-A., "A survey of visualization construction user interfaces," in [*Short Paper Proceedings of the Eurographics Conference on Visualization (EuroVis)*], (2013).

[26] Schulz, H.-J., Nocke, T., Heitzler, M., and Schumann, H., "A design space of visualization tasks," *IEEE Transactions on Visualization and Computer Graphics* **19**(12), 2366–2375 (2013).

[27] Kerracher, N., Kennedy, J., and Chalmers, K., "The design space of temporal graph visualisation.," (2014).

[28] Schulz, H.-J., Akbar, Z., and Maurer, F., "A generative layout approach for rooted tree drawings," in [*Pacific Visualization Symposium (PacificVis), 2013 IEEE*], 225–232 (2013).

[29] Schulz, H.-J., Hadlak, S., and Schumann, H., "The design space of implicit hierarchy visualization: A survey," *IEEE Transactions on Visualization and Computer Graphis* **17**(4), 393–411 (2011).

[30] Lü, H. and Fogarty, J., "Cascaded treemaps: examining the visibility and stability of structure in treemaps," in [*Proceedings of Graphics Interface 2008*], 259–266, Canadian Information Processing Society (2008).

[31] Pusara, M. and Brodley, C. E., "User re-authentication via mouse movements," in [*Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security*], 1–8, ACM (2004).

[32] Devroye, L. and Kruszewski, P., "The botanical beauty of random binary trees," in [*Graph Drawing*], 166–177, Springer (1996).

[33] Band, Z. and White, R. W., "PygmyBrowse: a small screen tree browser," in [*CHI'06 extended abstracts on Human factors in computing systems*], 514–519, ACM (2006).

[34] Blanch, R. and Lecolinet, E., "Browsing zoomable treemaps: Structure-aware multi-scale navigation techniques," *IEEE Transactions on Visualization and Computer Graphics* **13**(6), 1248–1253 (2007).

[35] Huang, M. L., Huang, T.-H., and Zhang, J., "TreemapBar: Visualizing additional dimensions of data in bar chart," in [*Information Visualisation, 2009 13th International Conference*], 98–103, IEEE (2009).

[36] Otjacques, B., Noirhomme, M., Gobert, X., Collin, P., and Feltz, F., "Visualizing the activity of a web-based collaborative platform," in [*Information Visualisation, 2007. IV'07. 11th International Conference*], 251–256, IEEE (2007).

[37] Hayasaka, K., Gojobori, T., and Horai, S., "Molecular phylogeny and evolution of primate mitochondrial DNA," *Molecular Biology and Evolution* **5**(6), 626–644 (1988).

[38] Saitou, N. and Nei, M., "The neighbor-joining method: A new method for reconstructing phylogenetic trees," *Molecular biology and evolution* **4**(4), 406–425 (1987).

[39] Hamming, R. W., "Error detecting and error correcting codes," *Bell System technical journal* **29**(2), 147–160 (1950).

[40] Elias, I. and Lagergren, J., "Fast neighbor joining," in [*Automata, Languages and Programming*], 1263–1274, Springer (2005).

[41] Evans, J., Sheneman, L., and Foster, J., "Relaxed neighbor joining: a fast distance-based phylogenetic tree construction method," *Journal of molecular evolution* **62**(6), 785–792 (2006).

[42] Sheneman, L., Evans, J., and Foster, J. A., "Clearcut: a fast implementation of relaxed neighbor joining," *Bioinformatics* **22**(22), 2823–2824 (2006).

[43] Bachmaier, C., Brandes, U., and Schlieper, B., "Drawing phylogenetic trees," *Algorithms and Computation* , 1110–1121 (2005).

[44] McKay, M., Beckman, R., and Conover, W., "A comparison of three methods for selecting values of input variables in the analysis of output from a computer code," *Technometrics* **42**(1), 55–61 (2000).

[45] Vliegen, R., van Wijk, J. J., and van der Linden, E.-J., "Visualizing business data with generalized treemaps," *IEEE Transactions on Visualization and Computer Graphics* **12**(5), 789–796 (2006).

[46] Bruls, M., Huizing, K., and van Wijk, J. J., "Squarified treemaps," in [*VisSym 00: Joint Eurographics-IEEE TCVG Symposium on Visualization*], 33–42, The Eurographics Association (2000).

[47] Cuadros, A., Paulovich, F., Minghim, R., and Telles, G., "Point placement by phylogenetic trees and its application to visual analysis of document collections," in [*Proceedings of the IEEE Symposium on Visual Analytics Science and Technology (VAST)*], 99–106 (2007).