Case Study: Visualizing Sets of Evolutionary Trees

Nina Amenta Jeff Klingner *
University of Texas at Austin

Abstract

We describe a visualization tool which allows a biologist to explore a large set of hypothetical evolutionary trees. Interacting with such a dataset allows the biologist to identify distinct hypotheses about how different species or organisms evolved, which would not have been clear from traditional analyses. Our system integrates a point-set visualization of the distribution of hypothetical trees with detail views of an individual tree, or of a consensus tree summarizing a subset of trees. Efficient algorithms were required for the key tasks of computing distances between trees, finding consensus trees, and laying out the point-set visualization.

1 Introduction

Systematic biology: Systematic biologists study evolution and construct evolutionary trees, also called *phylogenies*. Although they expect that all life belongs to a single evolutionary tree, its specific topology and even most of its details (e.g. the evolutionary tree for frogs) are far from clear. The most common methods for constructing phylogenies are based on DNA and RNA sequence data and are heavily computational. As more specimens are collected, sequencing methods improve, and algorithms and computer systems become more sophisticated, the datasets used in phylogenetic reconstruction become larger and more complex.

Constructing reliable phylogenies is important. They are used to answer questions like, "Where did human beings originate?", and, "Which strain of anthrax is this?". More fundamentally, evolutionary trees provide a framework in which other biological knowledge can be organized, and are used by biologists to search for pharmaceuticals, evaluate the threat of an invasive species to an environmental niche, and so on.

Application: The datasets involved in the process of computing phylogenies present a number of visualization challenges; our system addresses one in particular. Typically the genetic sequence data for a group of species or organisms (the *taxa*) are used as input to an optimization program that searches for the evolutionary tree that best explains the data, generally by "walking" from tree to tree. Whatever optimality criterion is used to define the "best" tree (maximum parsimony and maximum likelihood are two popular choices), the optimization program typically returns not one tree, but a set of hundreds or thousands of near-optimal or equally-optimal trees.

Traditionally, this rich and expensive dataset of optimal trees has been summarized with a single *consensus tree* and then thrown away. The consensus tree presents the features shared by all or most of the trees. If all the trees agree that a group of taxa have a common ancestor, a node corresponding to that common ancestor appears in the consensus tree. A consensus tree which is *well resolved* (that is, with many internal nodes, so that the tree is nearly binary) shows that the original data strongly support a single hypothesis about the evolutionary relationships of the taxa. If, on the other hand, the set of trees contains subsets which differ considerably from each other, the consensus tree of the whole set will contain very few internal nodes, all of high degree; in the extreme case, all of the taxa are connected to a single common ancestor and the consensus tree conveys no information. Our tool allows the biologist to interactively visualize and explore the whole set of trees, providing insight into the overall distribution and possible conflicting hypotheses.

We have also found that our system is useful for exploring sets which include non-optimal trees. In particular, biologists are interested in monitoring the computational search for optimal trees, to better understand the heuristics and perhaps to enable computational steering. Our tool can be used to visualize the search process, after-the-fact, as in Figure 2; we added coloring and animation features specifically for this application.

System Overview: As the visual interface, our tool provides two linked views of the data. The *point-set view* shows the overall distribution of trees. Each tree is represented as a point. The layout is computed using multidimensional scaling (MDS), a heuristic which places points so that the distances in the image reflect, as well as possible, the distances between the corresponding trees. We use a distance metric on trees which is related to the definition of the consensus tree, so that a set of points which are close together in the point-set view tends to have a consensus tree which is well-resolved. In many of the datasets we have examined, groups and clusters of trees in the point-set view produce well-resolved consensus trees.

The linked *tree views* are used to attach meaning to points or groups of points. When the user selects a single point, the corresponding tree is displayed. When a group of points is selected, the consensus tree is displayed. Selecting multiple points or groups allows for side-by-side comparison of the trees.

As an example, in Figure 1, the fact that there are three distinct groups of trees is apparent. The consensus tree of the small selected set of trees, on the left, is much less re-

*Computer Sciences Department, Austin, TX 78712, USA. Supported by NSF/DEB-0121682, and a Sloan Foundation Research Fellowship. amenta@cs.utexas.edu

solved than the tree on the right, belonging to the larger selected cluster, indicating that the larger cluster in some sense gives a more coherent evolutionary hypothesis.

Previous work: Our system is the first to address this application, but it includes existing elements from information visualization and computational biology.

Low-dimensional visualization of point-sets in high-dimensional or other difficult metric spaces, in particular by using MDS, is a textbook technique [3]. We were particularly inspired by the interface in the general-purpose MDS tool `XGvis` [4], which we used in initial experiments. Tree visualization is another central information visualization topic [15, 12], although systematic biologists use specialized programs for drawing evolutionary trees [11, 13]. Linking of the two views, including the computation of consensus trees on the fly, was needed to convey the meaning of the distribution. A design based on multiple linked views, with interaction via brushing [2], has worked well in other applications [9, 5].

Since efficiency is essential, we use a linear-time algorithm due to Day [8] for computing consensus trees and inter-tree distances. To our knowledge ours is the first implementation. We use sampling to speed up MDS [6]. In combining these elements, we chose to integrate MDS into an existing platform for phylogenetic computation and visualization, rather than adding the computational biology elements to an existing general-purpose MDS tool such as `XGvis`. We use the `Mesquite` platform [10], which provides tree visualization and other application-specific functionality. More importantly, it makes our tool convenient for biologists to find and use.

2 System design decisions

We tried to choose elements in our design, such as the distance metric and consensus tree definition, which are familiar and meaningful to the biologists as well as computationally efficient.

Evolutionary Trees: The leaves of an evolutionary tree are always labeled with the taxa, and permuting the labels on a tree with fixed topology generally produces a very different evolutionary tree. Internal nodes—the hypothetical ancestors—are generally unlabeled. Evolutionary trees may be rooted or unrooted, and edges may be weighted (with positive weights) or unweighted. Order is unimportant; for example, for a node in a rooted tree, swapping the left and the right child does not change the tree.

Computational biologists find it useful to represent evolutionary trees in terms of *bipartitions*. Removing an edge e_i from a tree separates the leaves on one side from the leaves on the other; this division of the leaves into two subsets is the bipartition b_i associated with edge e_i. An evolutionary tree is uniquely and completely represented by its set of bipartitions.

Consensus tree: Biologists define several kinds of consensus tree, including strict consensus, semi-strict consensus, and majority tree. Since we need to compute the consensus tree whenever the user brushes a new set of points, efficiency was our major decision criterion, and the strict consensus is the simplest to compute. The strict consensus tree of a set T of trees includes only those edges whose bipartitions are included in *all* the trees of T. The linear-time algorithm below computes the strict consensus tree in time $\Theta(nm)$, where n is the number of taxa and m is the number of trees. This is optimal since just reading m trees on n taxa requires $\Theta(nm)$ time.

Metric on Trees: There are several metrics on trees of interest to biologists; we use the Robinson-Foulds distance [14] (RF-distance). Like the strict consensus tree, RF-distance is defined in terms of bipartitions; hence trees close together with respect to RF-distance tend to have a well-resolved strict consensus tree. In addition, computing most alternative distances, such as nearest neighbor interchange distance (NNI) or tree bisection and reconnection distance (TBR), is NP-complete [7, 1].

The RF-distance between two trees counts the number bipartitions that are *not* shared by the two trees:

$$\frac{|\{b_i \in T_1, b_i \notin T_2\}| + |\{b_i \notin T_1, b_i \in T_2\}|}{n}$$

The distance is normalized by n, the number of taxa. Since RF distance is a particular kind of Hamming distance, it is a metric.

Layout: The point-set visualization is done with multi-dimensional scaling (MDS), a technique which takes a matrix of dissimilarities $\delta_{i,j}$ as input and produces a layout of points in a low-dimensional space $\mathbb{R}^k$, attempting to make the the Euclidean interpoint distances $d_{i,j}$ in the layout reflect the $\delta_{i,j}$ as well as possible. We currently lay out the point distribution only in $\mathbb{R}^2$.

The other option we considered is Principal Components Analysis (PCA), which computes a linear mapping of the points into the $\mathbb{R}^k$. The mapping produced by PCA is guaranteed to minimize $\sum_{i,j} | \delta_{ij}^2 - d_{i,j}^2 |$. This tends to weight large dissimilarities more strongly than one might like. With MDS we are free to choose a more natural optimization function, so as to produce better layouts. We used the popular *Stress* or *Kruskal-1* function [3], which is equivalent to optimizing $\sum_{i,j}(\delta_{i,j} - d_{i,j})^2$. The drawbacks of MDS are that it is slow and that it may get stuck in local minima.

3 Efficient algorithms

Efficiency is essential to a successful interactive experience. The three computational bottlenecks in our system are the computation of the inter-tree distance matrix, the layout of points using MDS and the computation of consensus trees in response to user queries. The brute-force algorithms for these problems are all $\Omega(n^2)$; MDS is an iterative algorithm which is $\Theta(n^2)$ per iteration. We require interactive performance on sets of thousands of trees and hundreds of taxa. To achieve this, we implemented a linear-time algorithm from the computational biology literature for the distance matrix and consensus tree computation, and we used sampling to reduce the time per iteration of MDS.

Consensus trees: The strict consensus tree for a set of n trees can be computed by finding the consensus tree T for the first two, finding the consensus tree of T and the third, and so on; hence we focus on finding the consensus tree for two trees. The brute-force algorithm represents the two trees by their sets of bipartitions and compares the two

sets. Although there are only a linear number of bipartitions in each tree, writing down a bipartition in the obvious way requires $\Theta(n)$ time, so overall the brute-force approach requires $\Theta(n^2)$ time. Day's linear-time algorithm [8] improves on this by using a sufficiently specific constant-size representation for bipartitions. We first index the taxa based on their order in a traversal of the first tree. A subtree corresponding to a bipartition is then represented by the minimum index i_{min} and maximum index i_{max} contained in the subtree, as well as the number of leaves l in the subtree. For a bipartion in the first tree, $l = i_{max} - i_{min} + 1$, so it is the only bipartition given this representation. Representations for which $l < i_{max} - i_{min} + 1$ may be used by many possible bipartitions, but no such bipartition can belong to the consensus tree since it cannot belong to the first tree. Given this representation, the algorithm is simply to store the bipartitions of the first tree in a dictionary, and look up the bipartitions of the other tree.

Distance computation: The algorithm for computing the RF-distance between two trees is very similar. Since we need to compute the distances between all pairs of trees in the input set, we can arrange for some additional efficiencies. The dictionary for each tree is computed only once, and each tree needs to compute its distance only to trees following it in the file. Nonetheless, computing the distance matrix, which is done once on start-up, requires $O(nm^2)$ time, for m trees and n taxa.

Linear-iteration MDS: Each iteration of MDS adjusts each point's position slightly in a direction that decreases the layout's stress, requiring $\Theta(n^2)$ time.

Our approach to scaling up the algorithm is to optimize the low-dimensional layout with respect to only a fixed-size subset of the interpoint dissimilarities, on the assumption that the full dissimilarity matrix is generally very redundant. Certainly on our data, the layouts we computed based on a subset of dissimilarities resembled those computed using all the distances, e.g. in Figure 3. We tried several ways of selecting this subset of dissimilarities. The simplest was to select a random subset of the points as an *anchor set* Ψ, so that only dissimilarities $\{d_{ij} : p_i \in \Psi\}$ are considered. A second method takes into account only the k largest and k smallest dissimilarities involving each point, for some fixed k. With this method, the layout's overall structure is outlined by the large dissimilarities, and the the fine detail of each point's position is determined by its near neighbors. Our third method, similar to that of [6], uses all the dissimilarities among points in the anchor set as well as the biggest and smallest dissimilarities for each of the rest of the points. In addition, membership in the anchor set is reassigned randomly for each new iteration of MDS, so that over time all dissimilarities among points are eventually taken into account.

We evaluated the performance of these different sampling methods by comparing the layouts they generated against those produced by the original MDS algorithm. We looked at both the increase in stress for the sampled-MDS layouts as well as the degree to which they were distorted with respect to the full-MDS layouts, where we define distortion as the average error in inter-point distances between two layouts: $\sum_{i,j} |d_{1_{ij}} - d_{2_{ij}}|$, divided by $n(n-1)/2$, the number of interpoint distances. Of our three sampling techniques, we found that the third method gives the best tradeoff between layout quality and the speed of MDS.

4 Acknowledgments

Three students from Lehman College of CUNY were responsible for recent improvements to this package. Fred Clarke and Denise Edwards helped with an internal redesign, while Silvio Neris implemented the consensus tree algorithm. We gratefully acknowledge their contributions.

We thank our biology collaborators David Hillis, Derrick Zwickl, Robert Jansen and Beryl Simpson, and our computer science collaborators Tamara Munzner, Tandy Warnow and Katherine St. John, for many ideas and discussions.

References

[1] B. Allen and M. Steel. Subtree transfer operations and their induced metric on evolutionary trees. *Annals of combinatorics*, in press.

[2] R.A. Becker and W.S. Cleveland. Brushing Scatterplots. *Technometrics* 29:127–142 (1987). Reprinted in Dynamic Graphics for Data Analysis, edited by W. S. Cleveland and M. E. McGill, Chapman and Hall, New York, 1988.

[3] I. Borg and P. Groenen, *Modern Multidimensional Scaling*, Springer, 1997.

[4] A. Buja, D. F. Swayne, M. Littman, N. Dean, H. Hofmann, XGvis: Interactive Data Visualization with Multidimensional Scaling, *Journal of Computational and Graphical Statistics*, to appear.

[5] A. Buja, D.F. Swayne, and D. Cook. Interactive high-dimensional data visualization. *Journal of Computational and Graphical Statistics*, 5(1):78–99, 1996.

[6] M. Chalmers, A linear iteration layout algorithm for visualizing high-dimensional data. In *Proc. of IEEE Visualization*, 1996, pages 127-132.

[7] B. dasGupta, X. He, T. Jiang, M. Li, J. Tromp, and L. Zhang. On distances between phylogenetic trees. In Proceedings of the 8th ACM-SIAM Symposium of Discrete Algorithms, pages 427-436, 1997.

[8] W. Day, Optimal algorithms for comparing trees with labeled leaves *Journal of Classification* 2:7-28. 1985.

[9] S.G. Eick and G.J. Wills. Navigating Large Networks with Hierarchies. In *Proc. of IEEE Visualization*, 1993, pages 204–210.

[10] W. P. Maddison and D.R. Maddison. Mesquite: a modular system for evolutionary analysis, version 0.96 beta. `http://mesquiteproject.org`

[11] D.R. Maddison and W. P. Maddison, MacClade version 4: Analysis of phylogeny and character evolution, Sinauer Associates, Sunderland Massachusetts, 2000.

[12] T. Munzner, Drawing large graphs with H3Viewer and Site Manager. In *Proceedings of the Symposium on Graph Drawing*,1998, pages 384–393.

[13] Page, R. D. M. TREEVIEW: An application to display phylogenetic trees on personal computers. *Computer Applications in the Biosciences* 12:357–358, 1996.

[14] Robinson, D.F. and L.R. Foulds. Comparison of phylogenetic trees. *Mathematical Biosciences*, 53:131–147, 1981.

[15] B. Shneiderman. Tree visualization with tree-maps: A 2-D space-filling approach. *ACM Transactions on Graphics*, 11(1):92–99, January 1992.

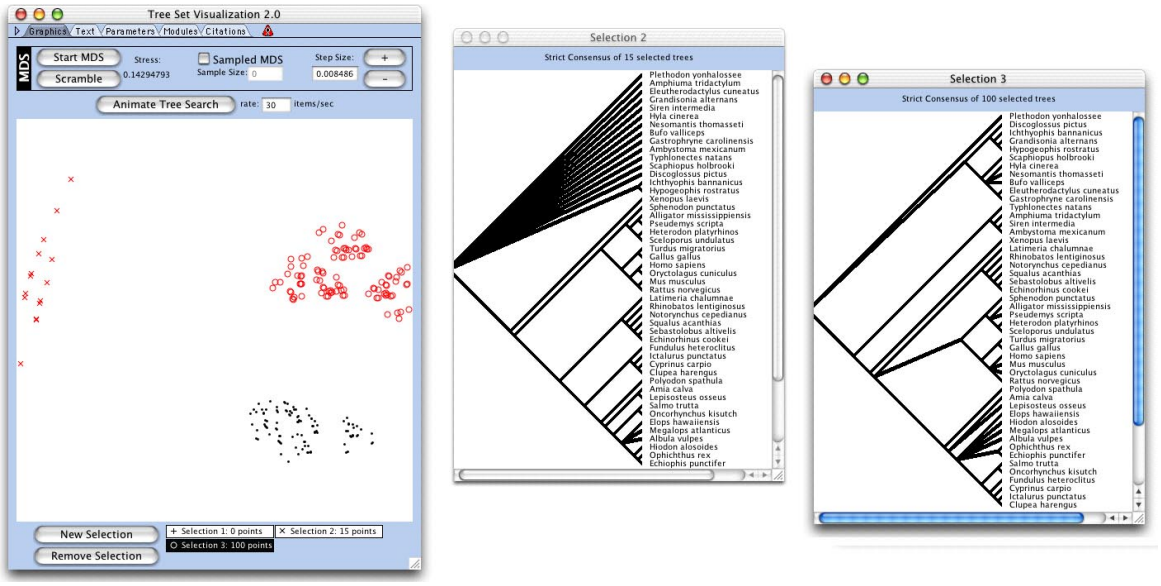

Figure 1. A sceenshot of our tree set visualization tool. On the left, the point-set layout of the distribution of trees. Each point represents a tree, and distances between points reflect distances between trees in an appropriate metric. Two subsets of trees are selected, and their consensus trees appear in the windows on the right. A consensus tree includes only the branches which are agreed upon by all of the trees in the subset. The dataset is a collection of optimal evolutionary trees for a group of animals, computed from RNA sequence data.

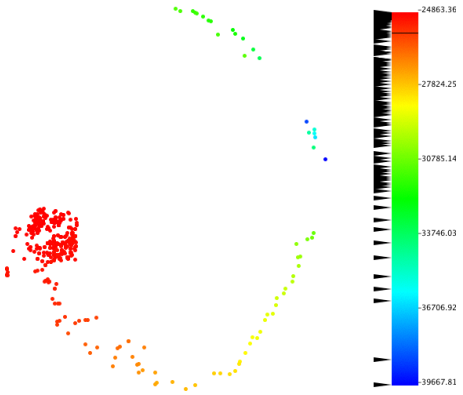

Figure 2. A layout in which points are colored by the optimality of the corresponding trees. The distribution of optimality scores is shown by the arrows on the scale bar to the right, and the black line represents the mean. The dataset consists of a regular sample of the trees computed during a Markov-Chain Monte-Carlo search for a maximum liklihood tree for New World frogs. An animation mode adds points to the display in the order in which the corresponding trees were computed, displaying the progress of the heuristic search process.

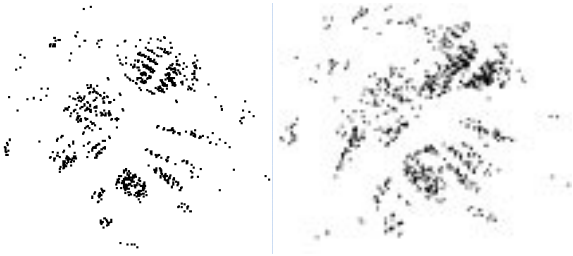

Figure 3. On the left, a layout given by normal MDS, using all of the interpoint dissimilarities at each iteration. On the right, the same points are arranged using only the dissimilarities involving a randomly selected anchor set of 100 out of the 1000 points. This layout is distorted but can be computed more quickly.