Aggregated Dendrograms for Visual Comparison Between Many Phylogenetic Trees

Zipeng Liu, Shing Hei Zhan, and Tamara Munzner *Senior Member, IEEE*

Abstract—We address the visual comparison of multiple phylogenetic trees that arises in evolutionary biology, specifically between one reference tree and a collection of dozens to hundreds of other trees. We abstract the domain questions of phylogenetic tree comparison as tasks to look for supporting or conflicting evidence for hypotheses that requires inspection of both topological structure and attribute values at different levels of detail in the tree collection. We introduce the new visual encoding idiom of aggregated dendrograms to concisely summarize the topological relationships between interactively chosen focal subtrees according to biologically meaningful criteria, and provide a layout algorithm that automatically adapts to the available screen space. We design and implement the ADView system, which represents trees at multiple levels of detail across multiple views: the entire collection, a subset of trees, an individual tree, specific subtrees of interest, and the individual branch level. We benchmark the algorithms developed for ADView, compare its information density to previous work, and demonstrate its utility for quickly gathering evidence about biological hypotheses through usage scenarios with data from recently published phylogenetic analysis and case studies of expert use with real-world data, drawn from a summative interview study.

Index Terms—Tree comparison, phylogenetic trees, level of detail.

1 Introduction

Biologists who study phylogeny, which aims to resolve evolutionary relationships among all living and extinct organisms, often find it difficult to make sense of large collections of phylogenetic trees. These evolutionary trees are typically computationally inferred from living organisms. The ability to gather genetic data quickly and cheaply has led researchers to generate and analyze more and larger trees [1], including intermediate trees that arise within a computational pipeline for reconstructing a final tree. They frequently need to inspect these trees to consider their plausibility and meaning with respect to existing domain knowledge. They do not blindly trust automatic computational metrics, especially when dealing with unknown genes and species or when developing new concepts and methods. Current visual analysis tools fall short of their need to make comparisons between large number of trees at multiple levels of detail.

In the visualization community, visual comparison of trees has been studied for years [2], but existing visualization techniques do not scale to the current needs of phylogenetic researchers. We characterize scalability of visual tree comparison in terms of three factors: the number of nodes per tree, the number of trees being compared, and level of detail at which the internal structure needs to be shown. Different tasks require very different levels of detail. These factors have separable requirements with respect to screen real estate and human cognitive ability: techniques that are suitable for scaling up in one of these may not support the others. For example, a technique for inspecting clusters made of hundreds of trees does not typically support fine-grained inspection at the subtree level within each one. Most scalable visual tree comparison work has been in support of comparison between a very small number of large trees. Approaches such as TreeJuxtaposer [3] are able to show highly detailed local structure, such as all descendants of a specific subtree. A few systems tackle the inverse problem of handling a large number of trees, where each tree is shown as a single point [4]. In this case, where a global distance measure is used to lay out those points, the available level of detail is an overview of the entire collection of trees. Between the two extremes, there is some work comparing dozens of trees at varying levels of detail, in many domains ranging from phylogenetic trees [5] to natural language processing [6].

Modern phylogenetic research takes place at a variety of scales: the number of trees can range from one to over ten thousand trees, and the number of nodes per tree ranges from a dozen to millions. The visual comparison of phylogenetic trees also encompasses a broad and complex spectrum of tasks. The research questions can range from assessing the overall uniformity or variance across a large collection of trees, to the exact placement of a particular leaf node across a small set of trees. Questions may pertain only to topological relationships between or within subtrees, or only to the distribution of attributes associated with nodes or branches, or to the combination of both. The problem space is huge; we worked with a group of phylogenetic researchers to understand their specific unmet needs.

We present ADView, an interactive system to visually compare one tree against hundreds of other trees, showing multiple levels of detail across multiple linked views. We propose an innovative technique to compress a tree into a concise representation that we call an **aggregated dendrogram (AD)**, which captures the major topological relationships between several subtrees that are the focus of user interest. Our best-effort layout algorithm yields an AD that can adapt its size and the amount of presented information according to the available screen space. We cluster similar ADs to generate a visual summary of topological relationships between chosen subtrees that reveals informa-

tion across multiple scales for datasets of hundreds of trees, enabling more detailed comparison than previous tools can achieve. ADView also includes a view showing distributions of trees in terms of their agreement and conflicts of leaf memberships for the chosen subtrees. It has an on-demand view for pairwise comparison in full detail, and auxiliary views for attribute values and other information. We show the results of using ADView on real-world datasets, including usage scenarios based on published datasets and case studies arising from an interview study with five domain experts.

In summary, the contributions of this paper are:

- task and data abstractions for the comparison of phylogenetic trees;
- an interactive visual tool to compare one phylogenetic tree against a collection of hundreds of other trees using multiple linked views;
- a technique to compress a dendrogram into a small resolution-aware visual representation showing topological relationships between subtrees of interest;
- validation through algorithm benchmarks, usage scenarios, formative expert feedback, and a summative expert interview study that provided case studies.

2 Related Work

We discuss the visual representation of trees in general, in the biology domain, the many approaches to tree comparison, and task abstractions for trees and networks.

2.1 General tree visualization

Tree comparison falls within the very active research area of general tree visualization. The treevis survey by Schulz [7] enumerates hundreds of general techniques to visualize single trees, but they do not suffice for our chosen problem of tree comparison, which is more complex than displaying one tree [8]. The space efficiency of many tree representations is studied by McGuffin and Robert [9]. We were inspired by their efforts when designing the aggregated dendrogram. Previous approaches to the compression and simplification of general trees and graphs according to their statistical properties [10], [11] do not suffice to support our targeted tasks for phylogenetic analysis such as preserving monophyletic groups. The DOITrees approach [12] dramatically simplifies trees and like ours is a focus+context technique that uses elision rather than distortion, but does not support these targeted phylogenetic analysis tasks. Pretorius et al. [13] propose a visual encoding for cell lineage visualization, but it is focused on different requirements such as emphasizing symmetry.

2.2 Biological tree visualization

Biologists routinely use domain-specific tools such as FigTree [14], iTOL [15] and Dendroscope [16] to visualize and annotate a tree during analysis, and to generate publication-ready figures, but these tools do not support comparison at all. Biologists who are capable of programming could use Python or R libraries such as DendroPy [17] and Ape [18]. These libraries are handy for flexible data manipulation such as trimming and computing an additional attribute, but fall far short of the kinds of visual comparison that ADView provides.

2.3 Visual tree comparison

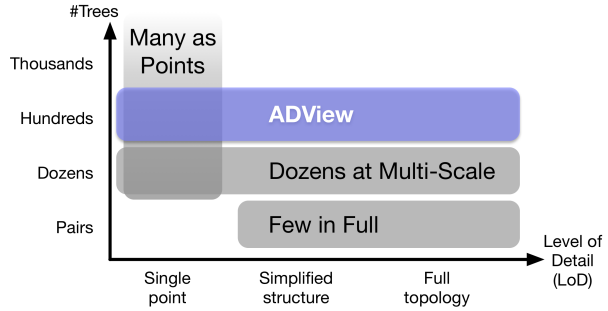


Fig. 1. The problem space of visual tree comparison for two scalability dimensions.

We categorize previous literature related to visual tree comparison by three dimensions of scalability: the number of trees, the level of detail at which the data must be shown to accomplish the intended task, and the number of nodes per tree. Fig. 1 illustrates the first two dimensions of this problem space. The vertical axis shows the *number of trees* partitioned into four categories: pairs, dozens, hundreds, and thousands. The horizontal axis shows the *level of detail* representing how many levels of detail a tool can show, in terms of three rough groups: a single point, simplified structure, and the full topology. We identify three major groupings: *Few in Full*, *Dozens at Multi-Scale*, and *Many as Points*.

The *Few in Full* category has been heavily studied. An extensive list of these systems appears the survey by Graham and Kennedy [2], and all of the entries in the InfoVis 2003 Contest on tree comparison fall into this category [19]. With only a small number of trees to handle, often just two, the most scalable previous systems can deal with a massive number of nodes per tree. For example, the TreeJuxtaposer system [3] supports detailed structural comparison of large trees up to several hundred thousand nodes. A similar system was recently re-implemented by Phylo.io as a web-based tool [20]. Some propose superposition to stack trees visually, such as the color coding of CandidTree [21] and the explicit encoding of Beck et al. [22]. Several systems proposed drawing lines connecting the same nodes on two trees [23], [24], [25]. Tree comparison is a special case of the more general problem of graph comparison. For example, Archambault proposes a pairwise comparison approach of hierarchically coarsening difference graphs [26]. These approaches by definition do not handle the larger collections of trees that we focus on in this work.

At the other end of the spectrum, the *Many as Points* systems handle hundreds or thousands of trees, with no limits on the number of nodes per tree. As our group name suggests, each tree is represented by a single point and thus these systems provide only a high-level overview of relationships between trees according to some kind of

similarity metric. Hillis et al. employ dimensionality reduction to map trees to a 2D space [4], revealing only cluster structure. Strimmer et al. propose Likelihood Mapping to visualize likelihoods for different hypotheses within a triangular coordinate system [27], but this tool is tailored for testing hypothesized relationships between four sequences or species families, and does not address more general problems. Fangerau et al. [28] compute a signature value for each tree and arrange them along a Hilbert curve to show structurally similar trees, and Hess et al. [29] cluster trees into a hierarchy to make sense of different parameter settings used in tree generation. The general idea of reducing complex structure to points is also applied to dynamic graph visualization by van den Elzen et al. [30]. These approaches cannot show the multiple levels of detail required for our target tasks.

The *Dozens at Multi-Scale* category comprises systems that handle from 2 to roughly 100 moderately-sized trees of around 100 nodes per tree, where at least two levels of detail are provided. Graham et al. propose a multi-view system for showing trees in different levels of detail using treemaps, node-link diagrams and lists [31], but they handle fewer trees than ADView and only a subset of tasks in Sec. 4. In DAViewer [6], Zhao et al. propose a compact view to aggregate information along two directions: from root to leaves, and for nodes across the same tree depth. Although it addresses questions about distributions and errors, it is not suitable for comparing topological structures at the subtree level and their choice of dendrogram does not scale to collections of hundreds of trees. PhenoBlock [32] uses small multiples to visualize many patient phenotypes compared to a single reference tree. Their solution takes advantage of the fact that phenotype ontology is fixed across all trees, so it would not work for the dynamic situations we present in Sec. 4. TreeVersity2 [33] embeds bar charts in trees to visualize changes of attribute values on nodes, while Vehlow et al. [34] and Bach et al. [35] deal with temporal changes of trees over time. All three of these systems support different data abstraction than ADView.

Consensus trees [36] are traditionally used by biologists to reason about collections of trees, where the places of agreement are shown as a standard tree and the disagreements are indicated by collapsing multiple leaves together under a single parent. Although they can be effective to pinpoint sparse disagreement in only a few areas, when substantial differences exist within a large collection of trees this simple approach breaks down. The summary graph of multiple domain specific graphs [37] proposed by Koop et al. has a similar limitation. DensiTree2 [38] uses half-transparent fuzzy branches to imply uncertainty in regions of difference, but this visual encoding shares the limitations of conventional consensus trees: fine-grained comparison is impossible when there is too much variation.

The previous work from Bremm et al. [5] is the most similar to ADView. They compare multiple phylogenetic trees at both global and local scales via topological summaries, for dozens of trees with dozens of leaves per tree. They also propose a new distance measure to capture structural differences between subtrees. Their approach for reducing visual complexity of trees is to retain the full path from a selected node all the way up to the root and hide the others. In contrast, our aggregated dendrogram method for simplifying trees, which preserves focal subtrees and carefully chosen local context while hiding more distant upstream nodes, is more compact and scalable. Moreover, our simplification approach can automatically adapt to available screen space. We present a detailed comparison of our results to theirs in Sec. 8.2, showing their limitations in handling for our identified tasks. In addition, they require all species under study to be present in all trees; in contrast, ADView can handle missing species, a situation faced by our domain experts that commonly occurs in phylogenetic analysis.

2.4 Task abstractions for trees

Gleicher offers some structural thoughts on the general task of visualizing comparisons [8]. The three papers devoted to task abstractions for networks only cover a subset of the tasks that we present in Sec. 4. Lee et al. [39], Pretorius et al. [40] and Kerracher et al. [41] cover tasks such as lookup by name, finding attribute ranges, and general questions about topology inspection. However, they do not address tasks such as finding corresponding branches or conflicting taxa across a tree collection, much less the questions very specific to phylogeny such as understanding monophyletic groups. Several previous papers that propose techniques for phylogenetic tree comparison do present abstractions for more specialized tasks, but none cover all seven of the tasks that we propose and none explicitly identify all five of the relevant levels of detail that we characterize in the data.

3 Phylogenetic Tree Data

We introduce some basic domain terms that pertain to phylogenetic tree data and present an abstract data specification.

3.1 Phylogenetic tree

A **phylogenetic tree** describes the evolutionary history of living organisms. Leaf nodes are called **taxa**, and are extant species that are being studied. Internal nodes represent common ancestors, usually inferred by statistical methods of phylogenetic reconstruction. Tree collections may arise from multiple reconstruction computations with different parameters or assumptions. A **subtree** is composed of all descendants beneath a branch; the set of all taxa underneath a specific branch forms a **monophyletic group**, also called a **clade**. There is a 1-to-1 correspondence between a branch, an interior node, a subtree, and a monophyletic group.

Tree collections also arise when inferring **gene trees** that represent the different evolutionary histories of individual genes; the synthesis of these multiple histories into a single overarching **species tree** requires extensive comparison between these trees.

3.2 Data abstraction

Phylogenetic researchers conduct comparison with respect to a single special tree that can be interpreted as their main hypothesis, where they are looking for evidence to support or invalidate the hypothesis from a collection of other trees. We denote the special hypothesized tree as the **reference tree** and the other trees as the **tree collection**. Both are

required as input data for ADView. Two typical scenarios are either one species tree vs. its associated gene trees, or one species tree vs. multiple species trees generated by different methods or parameters. The leaf nodes in these trees come with names, but internal nodes are usually not labeled.

Each tree branch has three attributes, two original and one derived. The primary original attribute is a **support value** associated with the branch, which is generated as an (un)certainty measure of how confident the reconstruction program is in its decision that the underlying leaf nodes should be grouped together. Each branch also has the secondary original attribute of *branch length*, which measures a distance from its ancestors related to evolutionary time. Although none of our target tasks directly rely on branch length, these lengths can provide useful context for biologists and are also sometimes used for sanity checking the data.

While the entire group of these trees has substantial overlap in the set of taxa, in many datasets an individual tree will have missing taxa. For example, in a collection of trees about the evolution of animals, some trees might not have kangaroos while others might not have turtles. We assume that trees are all rooted and binary. We also assume that every leaf node is unique in a tree; in biological language, that there are no *paralogs*.

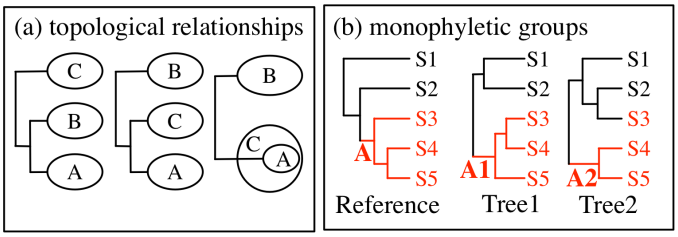


Fig. 2. Task Illustrations. (a) Three topological relationships among subtrees A, B, and C: A and C are separated, A and C are sisters, A is nested beneath C. (b) Tree1 agrees and Tree2 disagrees with Reference on the monophyletic taxa group (leaf set) under branch A. Reference tree branch A has the corresponding branches A1 and A2 in Tree1 and Tree2 respectively.

Many of the tasks presented in Sec. 4 require the derived data of **corresponding branches**: for a specific branch in the reference tree, we must find its counterpart of the most similar branch in each tree of the tree collection, as illustrated in Fig. 2(b). For example, the monophyletic group exploration task (T5 in Sec. 4) hinges on whether there is an **exact match** with similarity exactly equal to 1.0 between a reference tree branch and a corresponding branch, or an inexact match where the similarity is strictly less than 1. This computation is similar to the *best corresponding node (BCN)* proposed in TreeJuxtaposer [3]; algorithm details are provided in Sec. 7.1.

We use corresponding branches to also derive a third attribute, the *gene support frequency* (GSF), on every reference tree branch: the percentage of trees in the entire tree collection with exact matches for that branch. This metric captures uncertainty information of a subtree in the reference tree with respect to a tree collection, where support values usually fall short; several other biological metrics, such as the internode certainty [42], have similar goals.

4 Task Abstraction

The overarching goal in ADView is to compare a single reference tree to a collection of many other trees. We break down this goal into a set of seven specific tasks that were identified through multiple rounds of interviews with four biologists who work across multiple sub-fields of phylogenetic research including plant phylogeny, insect phylogeny, and mathematical modeling. We held weekly meetings with one of them, who specializes in plants and is a co-author of this paper, to allow frequent checks on the validity of our task abstractions and the utility of our visual interface during incremental refinement. In addition, we reviewed the biology literature on phylogenetic reconstruction and analysis.

In the language of Kerracher and Kennedy [43], we take a multi-strand approach to task gathering, primarily via interviews with domain experts and secondarily by deriving from literature. Two potential threats to validity did not apply in our case: the domain experts were indeed available, we were able to maintain a focus on task discussions, and skewing tasks towards a single domain is a feature not a bug given our goal to target phylogenetics in particular. We addressed the potential difficulties in introspection by conducting many rounds of elicitation and followup.

The seven tasks are:

- T1: Find subtree by **taxa names**. Biologists are typically familiar with the names of species in their study; an example is to find the subtree that consists of mammals.
- T2: Find branch / subtree with respect to an **attribute range** in the reference tree. The two attributes of support value and gene support frequencies are useful in suggesting interesting areas to investigate.
- T3: **Pairwise compare** one tree in the collection with the reference tree in detail. Allowing biologists to inspect a tree in full detail alongside with the reference tree allows them to make biological judgment calls about their findings. For example, they can see how species in a subtree are distributed in a target tree compared to the reference tree: whether there are any outlier species that are far away from others or scattered across the whole tree.
- T4: Inspect **topological relationships** between multiple subtrees. Fig. 2(a) shows three examples of relationships: nested, separated, or sisters. A subtree can be **nested** within another subtree. Otherwise, biologists would like to know if they are **separated** or direct siblings, known as **sisters**. They also want to know which subtrees are the closest to each other, and have a rough sense of the topological distance between specific subtrees.
- T5: Explore the **monophyletic groups** (the complete leaf set of a subtree) in terms of agreement or disagreement between trees, as illustrated in Fig. 2(b): do the taxa nested beneath a branch in one tree (the three leaves to the right of branch A) all fall beneath the corresponding branch in another tree? (They do for A1 in Tree 1 but not A2 in Tree 2.) How many and which trees agree with exact matches or

disagree with inexact matches? Do they agree with each other? How many alternatives are there?

- T6: Investigate the **conflicting taxa** between disagreeing subtrees. Users want to know which taxa cause the conflicts, allowing them to notice connections with previous findings such as whether a particular family of species is known to be hard to resolve.
- T7: Assess the **corresponding branch attributes** to associate their values with the (dis)agreement for monophyletic groups. This task is lower priority, but the uncertainty measure on the corresponding branches can help biologists to gauge their confidence on their findings. For example, a corresponding branch from a conflicting tree with an extremely low support value might signal a spurious conflict.

The actual workflow of biologists is incremental and iterative, so these tasks can occur in any order. We provide a few concrete examples of tasks sequences drawn from their analysis process, and discuss them further in Sec. 8:

- T1 $\leftrightarrow$ T2: sanity check on topology and attribute values of the reference trees, then identify interesting subtrees.
- T4 $\leftrightarrow$ T3 $\leftrightarrow$ T7: explore topological relationships, compare hypotheses, and support values distribution.
- T5 & T6 $\leftrightarrow$ T3 $\leftrightarrow$ T7: check agreement and conflicts of taxa memberships, inspect details of a specific tree, check support values.

These tasks led us to identify five relevant levels of detail: 1) tree collection, 2) a subset of trees, 3) individual tree, 4) subtree, and 5) individual branch or leaf node. These five levels provide a new abstraction that is a more precise articulation of the problem space than the three levels illustrated in Fig. 1, and informed the design of the Aggregated Dendrogram.

5 Aggregated Dendrogram (AD)

We propose a novel technique to aggregate tree representation that specifically addresses the difficult problem of topological relationships between multiple subtrees (T4) given hundreds of trees, and partially addresses the agreement of monophyletic groups task (T5). In this section, we indicate the motivation of the aggregated dendrogram, present its visual design, explain how similar ones are clustered, and share a glimpse of its evolution with a discussion of how earlier designs fell short.

5.1 Design requirements

In essence, the problem comes down to how to visually compress a tree with space requirements. First, the size of the compressed representation should be small to fit in as many trees as possible within a single screen. Second, the information contained in the compressed representation should adapt to the available screen space: given a larger area, it should convey more details than in a smaller one.

We determine what should and should not be compressed according to our task abstraction. Given a rooted tree and several focal subtrees, the critical information to show visually is the type of relationships between focal subtrees: either nested or separated and more specifically, (non-)sister relations (T4); also, whether the subtrees agree with the reference tree or not (T5). It can be helpful to also show distances between subtrees and more information about conflicting subtrees such as neighboring clades and placement of the conflicting taxa. Our design target is to support up to five focal subtrees, to provide a reasonable tradeoff between power and comprehensibility. Our investigation found that this number suffices for most phylogenetic questions. We design for a screen resolution of 1920x1080.

5.2 Visual design

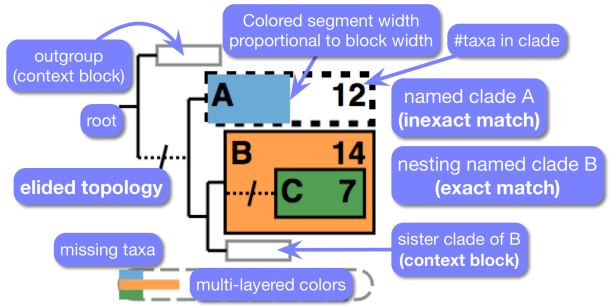


Fig. 3. Visual encoding of an aggregated dendrogram.

Driven by a combination of the design requirements and inspiration from previous work [5], [9], [12], we decided to take a non-distorting focus+context approach to compressing a tree, where less important information is elided [44, Chap. 14]. Specifically, we aggregate the dendrogram representation of a tree, as the biologists are most familiar with this form. Fig. 3 presents the final visual design of aggregated dendrogram (AD). In Sec. 5.4 we justify this design by discussing the shortfalls of several alternative designs that were confusing to the biologists; our task abstraction evolved iteratively, as the less successful attempts surfaced our misunderstandings by violating previously implicit assumptions of phylogenetic analysis.

A crucial design choice is to have each rectangular block in an AD always represent a subtree; that is, a monophyletic group in that tree. We also take care to visually distinguish whether each block is an exact match that agrees with a subtree in the reference tree, or an inexact match indicating disagreement, through different border styling.

The focal blocks represent the subtree beneath this tree's branch that corresponds to a selected one from the reference tree. These blocks have thick black borders and labels and their height is linear to the number of taxa inside, subject to the hard legibility constraints discussed in Sec. 7.2. Context blocks are rendered as less salient small rectangles with a grey border.

All AD block colors are consistent with the reference tree, where the proportion of colors indicate the percentage of highlighted taxa in this subtree. Exactly matching blocks, which have solid borders, are always fully colored because they exclusively contain all taxa under the reference tree branch. Inexactly matching blocks, which have dashed borders, are sometimes partly colored due to other taxa

that have *intruded*, as shown in the partial blue block in Fig. 3. Sometimes color appears in non-focal blocks due to *escaped* taxa, namely matched taxa that are found outside the corresponding subtree, shown by the tiny blue and orange context blocks between blocks A and B in Fig. 4(c).

There is only one kind of block that is not a subtree: the oval that can appear at the very bottom of the AD represents all missing taxa, as shown in Fig. 3. Its rounded corners clearly distinguish it from the other rectangular blocks, and its border is dashed. The distribution of different highlight colors for taxa within the same block is shown with vertically layered rows.

Unimportant parts of the tree are collapsed, as indicated by the special slash marker on a branch. For example, in Fig. 3, many evolutionary events along the path from the root to A's direct ancestor are elided, as we see from the slash crossing the dotted branch. Users can specify a collapse threshold of topological distance from highlighted blocks to either their lowest common ancestor or their nesting block. For example, Fig. 3 was made with the threshold set to 2; block C is more than 2 levels deeper than B, so everything between B and C is hidden. This threshold is a soft constraint in the legibility criteria discussed in Sec. 7.2.

Our approach depicts the topological backbone relationships faithfully. In Fig. 3, the following information can be easily interpreted: A and B are parallel but not sisters (direct siblings); C is nested within B with distance greater than two; A, B and C are far from the root (distance > 2).

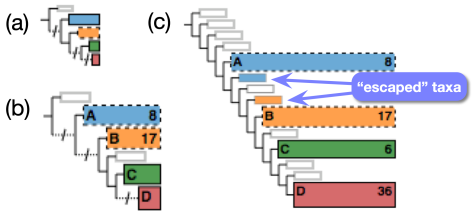


Fig. 4. Illustration of how the AD adapts to available space. These three ADs are based on the same tree and the same four focal subtrees, but with different size specifications: (a) 40, (b) 80, and (c) 160 pixels.

The AD algorithm, which we describe in detail in Sec. 7.2, adapts to available screen space automatically, as shown in Fig. 4. The algorithm makes a best-effort attempt to fit in as much information as possible, in terms of labels and context blocks, while keeping visible the critical information such as selected subtrees and their topological relationships (as defined in Sec. 5.1).

The individual ADs can be sorted by global distance to the reference tree, or by local distance with respect to a selected subtree.

5.3 Visual summary: Cluster AD

Although the individual ADs are spatially compact, when the tree collection is sufficiently large there can still be more than will fit within a single screen. We cluster the ADs that share the same topological relationships (as defined in task T4) and just render one representative AD, as illustrated in Fig. 5. By grouping ADs into clusters, we obtain a summary of all possible kinds of relationships among focal subtrees in the tree collection. The value of preserving individual

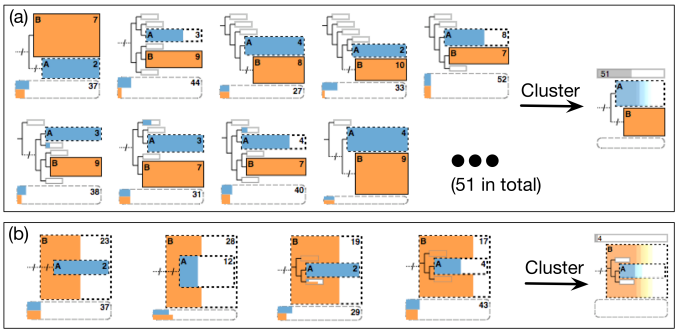


Fig. 5. Two examples of clustering structurally similar aggregated dendrograms. (a) Cluster groups 51 trees together where A and B are parallel. (b) Cluster groups 4 trees together where A is nested within B.

views that are topologically equivalent is they might end up highlighted differently depending on what other selections are made, which would provide useful information (as discussed in Sec. 6.6).

We ignore all context blocks while performing clustering, so that differences considered unimportant according to our criteria do not result in scattered or even spurious clusters. We randomly choose one of the constituent ADs from a cluster AD as a proxy for the entire cluster. The normal solid coloring is changed to a gradient to indicate the different proportions of filled color across the blocks in all ADs of that cluster. The clustering algorithm is presented in Sec. 7.3, and the gradient coloring is documented in Supplemental Sec.S2.

5.4 Design evolution

We tried many unsuccessful versions of aggregated dendrograms, four of which are shown in Fig. 6: remainder, container, fine-grained, and frond; it also shows the final skeleton layout which was successful. Eliciting feedback about these versions led to more clarity about what underlying biological meaning should be preserved in terms of both the focal and the context parts of the visual encoding.

One design question that required extensive iteration is the semantics for visible blocks in the diagram. Our first two attempts, the remainder and the container layouts, did not adhere to our current semantics that a rectangular block must represent a monophyletic group; that is, a complete subtree underneath a branch. We eventually realized that identifying monophyletic groups was an underlying subtask for the larger-scale biological questions that were being studied, so all alternatives where rectangles could represent non-monophyletic groups were confusing to the biologists. We then articulated T5 (monophyletic groups) as an explicit task during our iterative refinement of the task abstraction, which occurred in parallel with the design evolution.

The fine-grained layout did not scale past a few chosen subtrees. The frond layout also suffered from some scalability problems with excessive requirements for horizontal nesting, and moreover elided so much context that important questions were difficult to answer. The final skeleton layout was the result of several months of exploration in this very large design space of tradeoffs. Our final design features a careful mix of which parameters should be controlled by users, vs. hardwired, vs. adaptively computed.

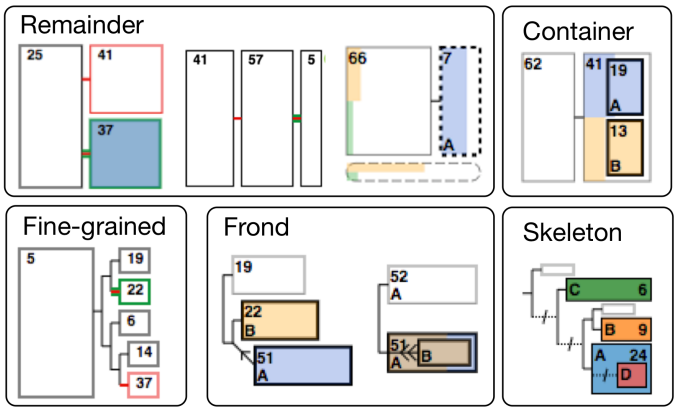


Fig. 6. Five versions of the aggregated dendrogram design in chronological order of invention. The first four were unsuccessful: remainder, container, fine-grained, and frond. The successful fifth layout, skeleton, built on lessons learned.

6 ADVIEW INTERFACE

We first give a brief tour of ADView, followed by the details about each of the views, and then discuss the view coordination in depth with respect to the five levels of detail.

6.1 Interface overview

Fig. 7 is a screenshot of ADView comparing a reference species tree against a collection of 68 species trees generated with different phylogenetic reconstruction methods. In the *Reference Dendrogram* on the left, four subtrees have been selected by the user to be the current focal groups, labelled A, B, C, and D and highlighted in blue, orange, green, and red, respectively. In the *Tree Distribution* view near the top, each row shows how the tree collection can be grouped by agreement or disagreement with respect to each focal subtree. Below that, the *Aggregated Dendrogram* (AD) view consists of cluster aggregated dendrograms and individual ones, showing topological relationships between the selected focal clades. Currently the user has selected the second cluster AD, highlighting 22 trees with a brown background, and is hovering on the second segment of row B in the *Tree Distribution* view, highlighting 12 trees with a black border. There are three auxiliary views showing additional information for the trees.

6.2 Reference Tree and AD Views

The *Reference Tree* view in the left pane of ADView is always devoted to showing the reference tree at full detail. We visualize this important phylogenetic tree with a branching style dendrogram with name labels for all taxa. This format is very familiar to biologists, intuitively supporting the Taxa Name task (T1). The horizontal length of a branch either encodes the branch length attribute, or uniform lengths can be chosen if high variance in branch lengths would lead to visual artifacts. In the *Reference Branch Attribute* view at the top of that pane, we show the distribution of the two main branch attribute values, the support values from raw data and the derived gene support frequency (GSF). The pink range selection area controls which branches that match the chosen attribute range are highlighted in pink in the *Reference Dendrogram*, in support of the Attribute Range task (T2).

The crucial interaction for ADView is to select a subtree in the reference tree and inspect the support and conflicts in the tree collection shown in the *Aggregated Dendrogram* views, in support of the Topological Relationships (T4), Monophyletic Groups (T5), and Conflicting Taxa tasks (T6). When the user picks an interesting subtree / branch, that subtree is labelled and colored in the *Reference Dendrogram*, a new highlighted block representing it is added to all of the aggregated dendrograms, and a row is added in the *Tree Distribution* view. ADView also allows for selecting a user-specified taxa group to handle cases where the user is interested in an arbitrary set of taxa that is not a monophyletic group in the reference tree, combining multiple choices using a popup menu rather than selecting a single reference-group subtree.

We provide a simple layer of automatic control for AD size that takes into account the number of trees that need to be drawn in each of the two views, the *Individual AD* and *Cluster AD*, to make intelligent use of the available screen real estate. The goal is to set a target size that preserves a reasonable amount of detail, ideally while fitting all of the ADs into a single pane without requiring vertical scrolling. When the number of trees is too large to fit even with small individual sizes, the leftover ones that do not fit within the pane are elided by default; the user can show them with the *More* button. The user can override this automatic size control by using a manual slider to indicate a specific desired height and width for all ADs. In both cases, the best-match AD layout algorithm is used to create layouts for the specified size. This size control mechanism was added late in the design process, after watching users miss opportunities to obtain better overviews.

6.3 Tree Distribution View

The *Tree Distribution* view at the top is focused on the Monophyletic Group (T5) and Conflicting Taxa (T6). Each row consists of multiple horizontal segments that represent groups of trees in the collection that agree with respect to the focal subtree, as shown in Fig. 8. A segment represents the subset of trees whose matching subtrees share the same taxa set with each other. The first segment shows the subset that agrees with the reference tree, which is marked with a circled *R* marker (R for reference). The following segments, sorted by frequency from left to right, are the conflicting subsets. Each subset indicates one hypothesis of the taxa set of a selected subtree, and thus the full row shows all alternatives of the taxa set that exist in the tree collection with respect to a specific branch.

When the user persistently selects or hovers over a segment, dot markers appear beside the taxa labels for that subtree in the reference tree dendrogram. In the top example of Fig. 8, the user hovers over the middle segment of row B, and we see from the black dots that the taxa set for this subset of trees does not have SPIROGYRA SP. In the bottom example, the user has persisted the dot markers for the middle and right segments in row A, and is now hovering on the left segment. The persistent selection triggers the reference tree labels to move from ragged-right layout to be

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2019.2898186, IEEE Transactions on Visualization and Computer Graphics

Fig. 7. Screenshot of ADView comparing a reference tree (left pane) against a tree collection (right pane).

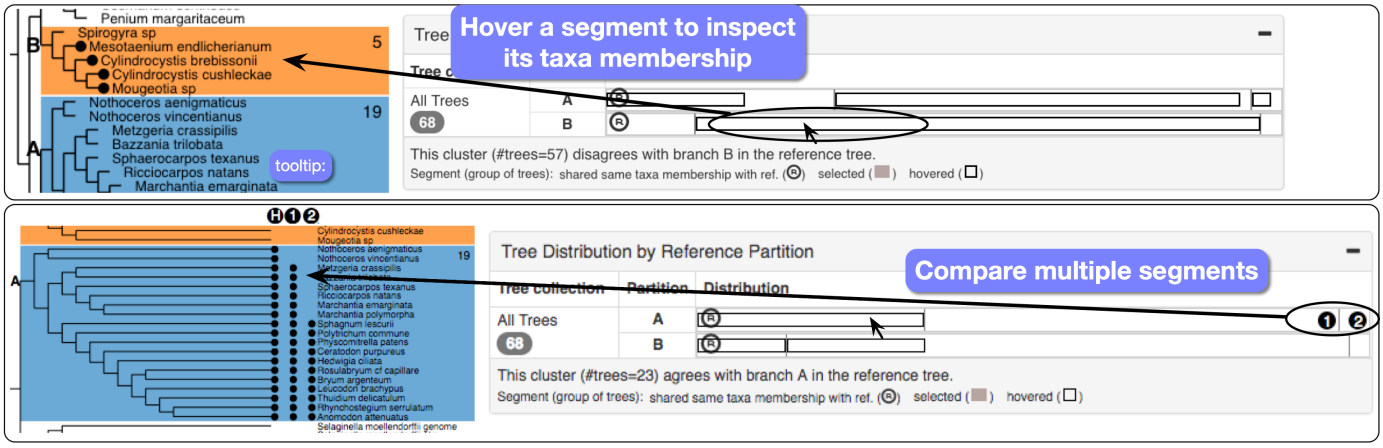


Fig. 8. Selecting a segment in the Tree Distribution view shows dots for each taxon within its group in the reference tree, temporarily in place on hover (above) or right-aligned for comparison between persistent selections (below).

right-aligned, to allow easy comparison of multiple columns of dot markers. The middle segment (labelled 1) has two taxa absent compared to the reference tree; that is, their closest subtrees to A do not include the first two taxa. The rightmost segment (labelled 2), which is very narrow because it only contains a few trees, is missing nearly half of the taxa in A.

6.4 Pairwise comparison

Users can select a specific tree from the tree collection for Pairwise Comparison (T3) with the reference tree in a **butterfly** layout, as shown in Fig. 9. This mirrored layout, where roots are on the outside and taxa labels are close to each other, is a familiar idiom for biologists. The highlighting is linked between the two trees, allowing users to locate taxa of interest in the target tree. This linkage provides a very salient way to quickly check whether a colored block selected from one side is contiguous (meaning that there is agreement between them), or dispersed on the other side (showing conflict for that selected subtree). The missing taxa are explicitly shown, and are integrated with the linked highlight color. This feature was added in a late iteration in the design process, after we realized the importance of providing awareness of specific missing taxa. The ability to inspect taxa membership from the tree distribution, as described in the previous section, is supported on both sides of the butterfly view.

Users can also select a subset of trees and create a consensus tree out of it (with extended majority rule [36]) to pairwise compare with the reference tree. This feature provides a convenient way to quickly get a sense of what

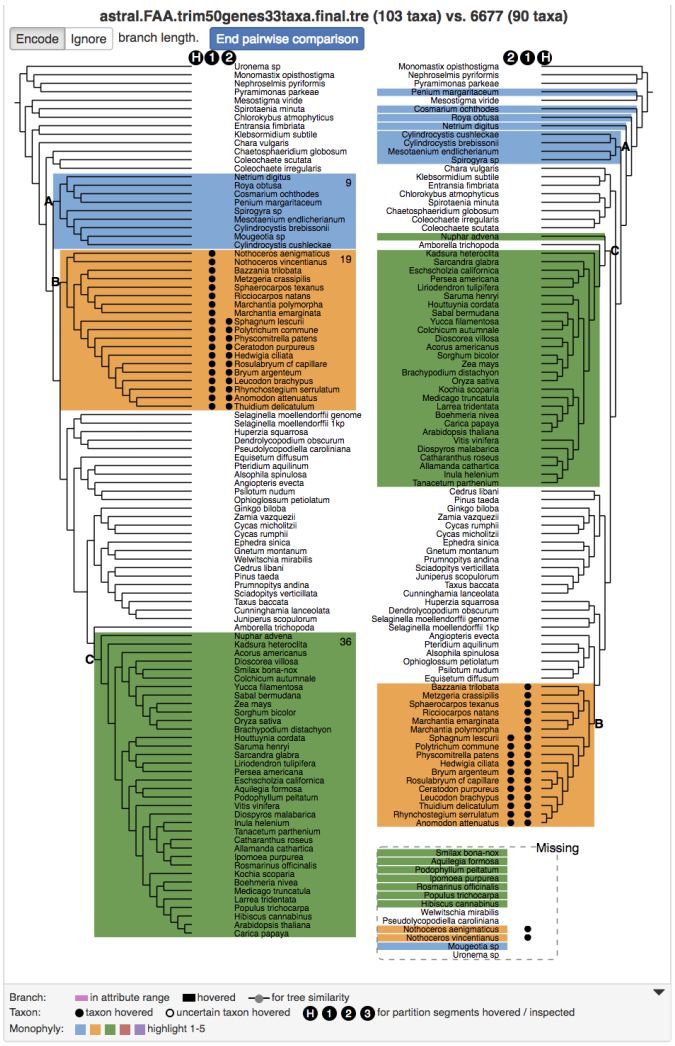


Fig. 9. Pairwise comparison between the reference tree (left) and a user-specified target tree from tree collection (right).

the subset of trees roughly looks like without diving into each tree one by one.

6.5 Auxiliary views

There are three auxiliary views on the far right.

The *Tree List* view at the top shows a list of names of trees with linked highlighting for the tree selection. Sometimes trees are named according to conventions that reveal useful information such as the data source, method, parameters to generate the tree, and gene identification numbers.

The *Tree Similarity* view in the middle encodes relative distances between pairs of trees to indicate potential cluster structure, where each dot represents a tree. This view is inspired by many papers in the category *Many as Points*. Distances between dots correspond to distances between trees as computed by the t-SNE dimensionality reduction technique [45]. The default distance metric is Robinson-Foulds to show global relationships, or the user can choose a the inverse Jaccard measure that provides local relationships with respect to a chosen subtree. These measures are discussed further in Sec. 7.1.

The *Corresponding Branch Attribute* view is aimed at the task of that name (T7) and shows the distribution of attribute values of corresponding branches for a selected focal branch / subtree.

6.6 View coordination and visual encoding

Visual indication of the coordination between multiple views is a complex and constrained problem in ADView because of the many levels of detail we support. Supplemental Table S1 provides a detailed breakdown of the most basic level of view coordination: what aspect of data is visually encoded (or not) across all levels of detail and all views. The size of visual elements varies enormously between the views: the region subtended by a single tree ranges from a small point in the *Tree Similarity* view to a medium-sized cell in the *AD* view; subtree blocks can be large in the *Reference Tree* view but small in the *AD* view.

This variability poses a difficult challenge for color coding because distinguishability depends on size. Nevertheless, given the power of color for showing categories and the desirability of color consistency across elements between views [46], we carefully designed a color palette in consideration of these tradeoffs. The five medium-saturation colors for the subtree highlight blocks are designed to be roughly equal luminance so that no focal subtree seems more important than the others, sufficiently salient for large regions in the *Reference Dendrogram* view and small regions in the *AD* view, against both the white default backgrounds and brown selected backgrounds. There is adequate luminance contrast for the black text label legibility, while maintaining distinguishability for the small highlighted pink branches that could occur within these blocks or against a white background.

We off-loaded some selection indicators to other visual channels, using additional point marks to show taxon selection in the *Reference Tree*, and border outlines showing tree selection for both points in the *Tree Similarity* view and cells in the *AD* views. Moreover, shape coding and line type coding is already used extensively in the *AD* design itself.

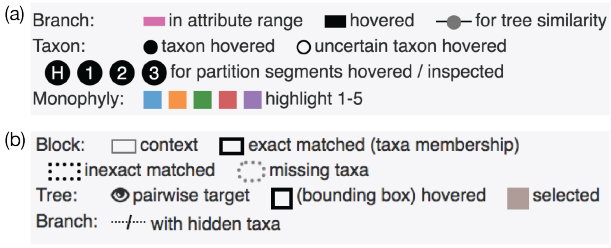


Fig. 10. Legends summarize the visual encodings. (a) Reference tree. (b) Aggregated dendrograms.

Fig. 10 shows a closeup of the two legends that appear at the bottom. To summarize, linked highlighting is encoded in several ways: five medium-saturation block colors for subtrees (blue, orange, green, red, purple), both background color (brown) and border outlines (black) to indicate two forms of tree selection, line color for branches and region color within attribute range histograms (pink), and additional point marks (black) for taxa.

7 ALGORITHMS

We provide an overview of the ADView architecture, then present specific algorithms in detail.

We use a browser-server architecture for ADView. The user can submit a dataset using the *Upload* page in browser, as shown in Supplemental Fig.S26 and Fig.S27, which sends the raw tree files (in Newick format [47]) to the server. A Python asynchronous pre-process worker in the server is then invoked as shown in Supplemental Fig.S28. It parses all trees, computes distances between all tree pairs and corresponding branches between each branch in the reference tree and each tree in the tree collection, stores them into a MongoDB database, and notifies the user upon completion. When a client session begins, the browser fetches data from a lightweight RESTful API, performs dimensionality reduction to the distance matrix using t-SNE [45], presents the data, and responds to user interaction such as updating the ADs when a new branch of interest is selected. We use React+Redux to manage state of data model, view, and user interaction in the browser, and we render almost all visual elements in SVG manually with some help from D3. The implementation is available as open source under an MIT license from https://github.com/zipengliu/adview.

7.1 Corresponding branches

Finding comparable counterparts between the trees, namely corresponding branches, is done in the server when the user uploads a new dataset. The **corresponding branch** is the most similar branch in each tree of the tree collection to a specific branch in the reference tree, analogous to the *best corresponding node (BCN)* proposed in TreeJuxtaposer [3]. An example is shown in Fig. 2(b).

We define the similarity between a branch b in reference tree R and a branch b' in a tree T of tree collection as the Jaccard index between two leaf nodes sets beneath the two branches, excluding missing leaves:

$$\frac{|b.taxa - T.missing| \cap |b'.taxa - R.missing|}{|b.taxa - T.missing| \cup |b'.taxa - R.missing|}$$

where $b.taxa$ and $b'.taxa$ are the leaf node sets under b and b', $R.missing$ and $T.missing$ are the respective missing leaf node sets. Missing taxa are excluded for a less biased comparison, following general practice in phylogenetics research. Failure to exclude would lead to a record of both conflicting and missing signals, in contrast to our goal which is to focus only on the conflicting signals.

This measure, which captures set memberships, is an intuitive extension to the standard biological convention of Robinson-Foulds distance metrics for trees [48]. This set-based measure is simple to compute and is a good match with our task abstraction and our visual encoding and interaction design choices. We note that Bremm et al. [5] propose a more complex similarity metric that captures both topological structure and set membership, which they use to create histograms that convey aggregate information about mid-level structure; however, the details about what is different cannot be seen with this approach because they are aggregated away. Our approach communicates a great deal of detailed mid-level topological structure through the ADs, so we only rely on the similarity metric for a first stage of exploring whether trees agree on what taxa are in a clade.

We need to compute the similarity score of every branch pair between the reference tree and tree collection, as shown in Algorithm 1. We use a bottom-up approach for the $GetSimilarity()$ function. Supposing $A = b.taxa - T.missing$ and $B = b'.taxa - R.missing$, the algorithm bottleneck is to compute $|A \cap B|$ when A is fixed. The leaf set of an internal node is comprised of the leaf sets of its children. We can thus compute $|A \cap B|$ by summing $|A \cap B_1|$ and $|A \cap B_2|$, where B_1 and B_2 are the leaf sets of B's left and right children excluding $R.missing$.

Algorithm 1: Corresponding branches to every reference-tree branch

Input : reference tree R, tree collection C
Output: all corresponding branches $corr$

```
for b ∈ R do
    for T ∈ C do
        targetSet ← b.taxa − T.missing
        for b′ ∈ T in post order do
            similarity ← GetSimilarity(b, R, b′, T)
            if similarity is better then
                corr[b][T] ← b′
            end
        end
    end
end

Function GetSimilarity(b, R, b′, T)
    if b′ is a leaf then
        // b′.card: cardinality of |b′.taxa − R.missing|
1       b′.card = 0 if b′.taxon ∈ R.missing else 1
2       b′.intersect = 1 if b′.taxon ∈ targetSet else 0
    else
        b′.card = b′.left.card + b′.right.card
        b′.intersect =
          b′.left.intersect + b′.right.intersect
    end
    union ← |targetSet| + b′.card − b′.intersect
    return b′.intersect/union
```

Algorithm analysis: Algorithm 1 computes similarity of all branch pairs between the reference tree and tree collection in $O(N^2 * |C|)$ time, where N is the number of leaf nodes in a tree and $|C|$ is the number of trees in the collection. Function $GetSimilarity()$ is amortized $O(1)$ as the checking statement in line 1 and 2 utilizes a hash table. The $O(1)$ similarity calculation is a substantial improvement compared to the $O(logN)$ TreeJuxtaposer [3] approach.

7.2 Aggregated dendrogram layout

Every time a subtree of interest in the reference tree is selected or removed, we update the layout of all aggregated dendrograms, which is handled on-the-fly by the browser. Given a tree data object, a set of focal subtrees, and the AD parameters, we use a best-effort mechanism to generate an AD layout that adapts to the specified size. The user-specifiable parameters exposed in the interface are the height and width of the AD, the number of context levels,

and whether to apply labels and colors. Several more algorithm parameters exist, and are illustrated in Supplemental Fig.S4. The block color is determined at a later step in our rendering pipeline; the layout algorithm described here only computes the position and sizes of blocks and branches.

Algorithm 2: Best-effort aggregated dendrogram layout

```
Input  : tree, focalSubtrees, params
Output: AD layout or failure
AD ← generateLayout(tree, focalSubtrees,
 params)
while not isLegible(AD) do
    params ← shrinkParams(params)
    if not params then
        return failureGlyph
    else
        AD ← generateLayout(tree, focalSubtrees,
         params)
    end
end
return AD
```

As shown in Algorithm 2, we keep generating tentative layouts with different parameter settings until the latest one passes a legibility test or parameter choices are exhausted. The function $generateLayout()$ compiles a list of blocks and then links them together with either normal or elided branches by a bottom-up traversal. The coordinates of blocks and branches are then calculated based on the block hierarchy and relevant AD parameters such as total width and height. The function $isLegible()$ takes a layout and determines whether the blocks and branches will be visible if drawn. The hard target is that focal subtrees need enough room to be colored; the soft target is to have enough room for labels as well, as a redundant encoding if space permits. A set of parameters will be "downgraded" from more sprawling to more compact by the function $shrinkParams()$ every time the legibility test fails until the lower limit of all parameters is reached. There are two cascading sets of flexible parameters that can be changed: 1) the number of context levels, a metric to control how many context blocks to show; 2) inter-block gaps, branch lengths, and block sizes. The function $shrinkParams()$ first shrinks the first set until lower limit is hit, then starts to shrink the second set. If legibility is not achieved, the algorithm returns an error, which causes the interface to show a failure marker to notify users that there is no hope of fitting an AD in such space, indicating that the size parameters should be turned up manually using the UI sliders to override the automatic size-control algorithm one layer above this layout algorithm.

One example of a failure case would be rendering 5 selected nested sub-trees in 30x30 pixels. The user could decide whether it would be more appropriate to select fewer subtrees (for example, if some uncleared selections were left over from previous analyses rather than being crucial for the current investigation), or simply to allocate more space to each AD and accept the tradeoff that all of the individual ADs do not fit within a single screen. While our automatic algorithms do quite a bit of work on behalf of the user, our strategy is a mixed initiative approach that relies on the judgment of the expert user.

Algorithm analysis: The number of trials per AD depends on both the characteristics of the input tree itself, the amount of context to show, and other size related parameters. Our empirical observation is that with the current settings of ADView this number is less than 10 and usually just 1 or 2. The computational time of this algorithm is hence that of the function $generateLayout()$, which is linear to the number of nodes in a tree.

7.3 Cluster aggregated dendrogram

Algorithm 3: Cluster topologically identical ADs

```
Input  : AD_layouts
Output: Cluster_ADs
strings ← serialize(AD_layouts)
Cluster_ADs ← group(sort(strings))
Cluster_ADs ← sortByFrequency(Cluster_ADs)
```

After generating AD layouts for all trees, we cluster them into groups of shared topological relationships among focal subtrees, as shown in Algorithm 3. The serialization of AD computes a string that looks very similar to the Newick file format [47], which maps topologically identical AD layouts to identical strings. The user can choose to further distinguish exact and inexact matches, and sister-group or separate relationships. For example, the AD in Fig. 3 is serialized into a string "$(A,(C)B))$". If exactness and sister-group relationship is considered, it becomes "$(A-,(C+)B+)/$", where $+$ and $-$ stand for exact and inexact matches, and $/$ and $|$ for non-sisters and sisters. We only consider focal subtrees and ignore everything else. After serialization, the clustering problem is transformed into a trivial one: grouping the same strings together.

Algorithm analysis: Because serialization only requires a traversal of AD blocks, and the number of blocks is usually very small (less than a dozen), the time complexity of Algorithm 3 relies on sorting short strings. It is thus $O(n \log n)$, where n is the number of ADs.

8 Results

We present empirical benchmarks for the computational performance of the algorithms, an informal comparison focusing on information density to the Bremm et al. [5] system, and a usage scenario illustrating several typical workflow steps for ADView. We briefly report on formative feedback from ten domain experts during the iterative design process, and then present a summative expert interview study on the utility of ADView including two case studies. The supplemental material includes multiple full screenshots of the first usage scenario, a second usage scenario with over 1300 gene trees compared to a single species tree, and also a video to show the look and feel of the system in action.

8.1 Computational performance

We deployed the server programs and database to a small server with two 2.20 GHz Intel Xeon E5-2650 CPUs and

2G RAM hosted by our university, and run the client on a Chrome browser on a 2012 Macbook Pro with a 2.50 GHz Intel i5 CPU and 8G RAM. For a dataset with 500 trees and about 100 leaf nodes per tree, created by sampling from the public 1KP dataset (as discussed in Sec. 8.3), it takes roughly 50 seconds to pre-process the data on initial upload and 10 seconds to load in browser at startup. The size of the compressed dataset is about 2MB. The user interaction in the browser is fluid when the number of trees is less than roughly 500. There is noticeable delay of about 1 to 2 seconds for datasets beyond this scale when all individual aggregated dendrograms are displayed, but immediate responsiveness can be restored by collapsing that pane. We used a caching technique to accelerate the rendering of AD in browser, which is detailed in Supplemental Sec.3.2.

8.2 Information density comparison

We directly compare ADView to its closest competitor, the tool developed by Bremm et al. [5], using their dataset of 33 trees and 32 taxa. An initial impression from ADView is that the trees seem very different from each other: very few branches in the reference tree are supported by majority. In the rest of this comparison we focus on the visual encoding and interaction scalability.

Supplemental Fig.S19 shows their system after adding 12 trees to the right panel and selecting one subtree of the reference tree, which is highlighted in pink. All matched nodes are highlighted across the trees, but their system only allows one focal subtree at a time. Without any manual filtering, many irrelevant parts are visible and the layout is visually crowded. Supplemental Fig.S21 shows the view after we filter out elements with similarity scores below 0.5. Substantial space is required for layout because the entire path between the internal nodes and the root is preserved.

Supplemental Fig.S20 shows ADView with the same dataset after we select three subtrees in the reference tree. (The paper's materials do not include enough information to replicate the selected groups, so we randomly chose subtrees to highlight.) The cluster ADs show the major topological relationships between the blue, orange and green clades (task T4), and the *Tree Distribution* view shows the agreement and conflicts on each of the selected clades in terms of taxa memberships (T5). These two views in ADView provide topological information in a more space-efficient way than the full dendrograms in Fig. S19, and with far more detail than can be gleaned from their heatmap and histogram views based on similarity scores. If the similarity scores are of direct interest, in ADView the user can use sorting by similarity, see distributions in the histograms, or hover on a specific branch in the *Pairwise Comparison* view.

From the screenshots, we can conclude two major advantages of ADView over their system. First, one of the important tasks, topological relationships between subtrees (T4), cannot be supported by the color coding and filtering mechanism of the similarity score they propose, while the ADs in ADView are more suitable for this task. However, a usability disadvantage of our system is that due to the broader scope of task complexity, it is more difficult to learn. Second, ADView can scale to much larger number of trees due to higher spatial efficiency of tree representation. We also note that they do not show taxa names on their interface, which is an essential piece of information for our biologists to interpret the display. However, for datasets such as this one with a small number of trees and small tree sizes, their system has the advantage of showing selectable taxa in all of the small-multiple tree views, which fit on a single screen side by side.

8.3 Usage scenario 1: 1KP pilot study

The 1000 Plants Consortium (1KP) gathered phylogenetic data for thousands of plants and published a pilot study about the origin and early diversification of land plants [49]. One of the main research questions is to determine the sister group of LAND PLANTS (LP for short).

The authors picked 103 representative plant species of major groups in STREPTOPHYTES, and produced 69 species trees by performing 69 different analyses; that is, 69 runs of the computation pipeline, spanning different statistical methods and parameter settings, different filtering criteria, and different kinds of sequences. They compared these trees with previous hypotheses to the research questions.

We took their published data and checked whether their research questions could be answered quickly with ADView. This usage scenario was created with the domain expert who is a co-author on this paper, who has close ties to the researchers who wrote the original paper. We report on one of these questions below, and present a second one in Supplemental Sec.S6. We randomly picked one tree as the reference tree, using the other 68 as the tree collection. We also confirmed that the same results were reproduced by picking another reference tree, where all but 1 of the 68 aggregated dendrograms remained very similar.

Correspondence of cluster ADs to known hypotheses: the previous hypotheses said land plants (LP) might be the sister of ZYGNEMATOPHYCEAE (ZYGN), CHARALES (CHAR) or COLEOCHAETALES (COL), as illustrated in Fig. 11(a). We selected and highlighted these four clades in the reference tree as our focus. ADView generated the ADs and clustered them into five groups (without considering exactness but with sister-group relationships), which can be regarded as five hypotheses that this dataset supports. We quickly found out that clusters C1 and C4 support the ZYGN-sister hypothesis, clusters C2 and C5 support CHAR-sister, but cluster C3 does not agree with any of the three (task T4).

Support values in C1 + C4 (ZYGN-sister trees): when we checked the distribution of support values (T7) by selecting the trees of C1 and C4 and created a sub-collection, as shown in Supplemental Fig.S7, there are trees with low support values. We can interpret this display as a counter-evidence to the ZYGN-sister hypothesis.

Conflicts in ZYGN: inspecting the *Tree Distribution* view in Fig. 7, we saw that all 68 trees plus the reference tree agree that LP, CHAR and COL are grouped together (monophyletic). However, a small portion of trees disagree with the reference tree about ZYGN, as shown in the hovered-over segment (T5). To investigate what triggered the disagreement (T6), we selected those trees by clicking on that segment, made a consensus tree out of them, and pairwise compared with the reference tree (T3), as shown in Fig. 11(b). It turned out that a species, SPIROGYRA, was not

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2019.2898186, IEEE Transactions on Visualization and Computer Graphics

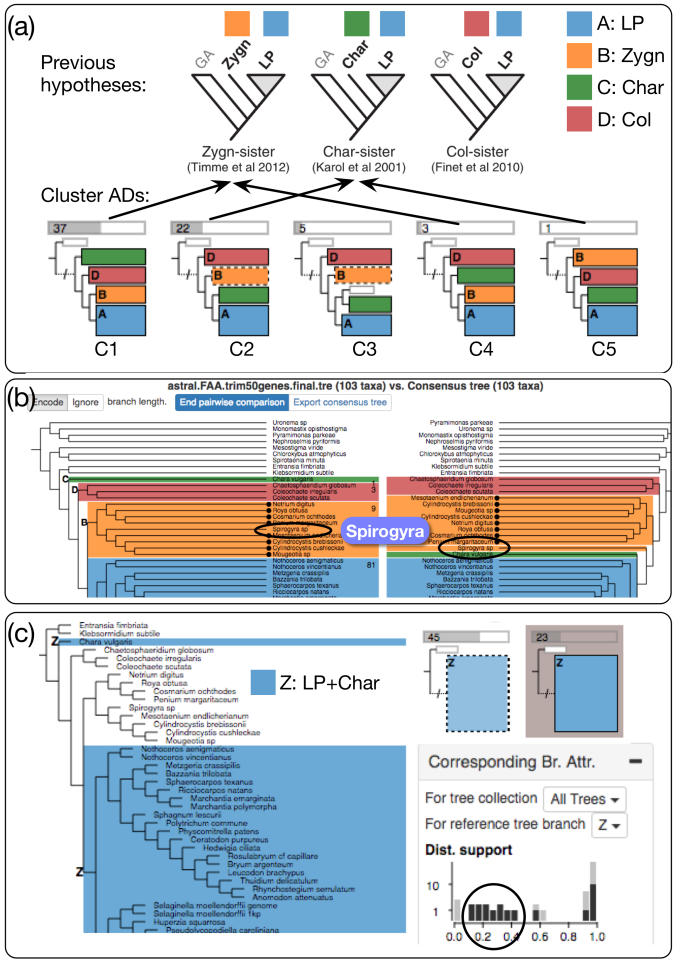


Fig. 11. Annotated screenshots for a research question in the 1KP pilot study: which is the sister group of LP? (a) Previous hypotheses (excerpt from their paper [49]) and the cluster ADs in ADView; arrows show hypotheses the cluster ADs support. (b) Consensus tree of cluster C3 pairwise compared to the reference. (c) Branch support values of trees supporting the LP+CHAR hypothesis.

included in ZYGN for these trees; we pairwise compared with each selected tree to double check (T3). The authors of the pilot study confirmed that there are complicated underlying issues for this misplacement.

Details in C2 + C5 (CHAR-sister trees): in Fig. 11(c), we further dug into the support for CHAR-sister hypothesis by combining LP and CHAR into group Z to see whether trees contain it (T1). Since Z is not a complete subtree in the reference tree we could not directly select it, so we used the popup menu that supports selecting an arbitrary set of user-specified taxa to choose the combination of LP and CHAR. . There were 23 exact-match trees, which is exactly the number of trees for cluster C2 plus that of cluster C5 (T4). In the *Corresponding Branch Attribute* view, we checked the black foreground bars in the histogram of support values for corresponding branches of these 23 trees (T7). The circled bars showed that some trees have very low support for LP+CHAR (biologists consider support less than 0.5 very low). The *Tree List* view showed that these trees are mostly generated from a kind of unfiltered sequence data that has been called into question in the domain literature.

The workflow description above covers some interesting and representative steps extracted from the full session, in which further visual analysis was conducted to check the supporting and conflicting evidence for additional hypotheses. By the end of the full session, we had found substantial evidence to support the pilot study's conclusion that ZYGN is probably the sister-group of LP, although there seems to be counter-evidence that merits further inspection.

8.4 Formative expert feedback

We obtained informal formative feedback during the iterative design phase from the four phylogenetic researchers who worked with us throughout the project. At a quite mature stage of development we also presented it to six other phylogenetic researchers via chauffeured demos. Overall we received positive feedback confirming that our task and data abstractions matched with their biological intuitions, and that the functionality provided by ADView would be helpful to them. We also gathered suggestions for improvement, many of which informed further iteration of the design.

8.5 Summative expert interview study

We conducted an expert interview study with five domain experts; our observations and their feedback served to confirm the validity of our task abstraction and interface design. We describe the participants and procedure, present two case studies here (with an additonal one in Supplemental Sec.S8.3), and discuss their feedback.

8.5.1 *Participants*

We recruited five participants who are actively involved in phylogenetic research and conducted four study sessions of 90 minutes each. In this paper, we refer them as P1, P2, P3, P4a and P4b (the latter two were in the same shared session). Four of them come from different labs at our own university, and one from overseas. Four were researchers focusing on biological questions with different kinds of organisms, three of them PhD students and one principal investigator; one was an experienced bioinformatician whose daily job is to manage, process, analyze phylogenetic data of his lab. Two had seen a previous version of ADView demonstrated to them a few months before the study took place; one had seen screenshots accompanied by a general description. Participants P4a and P4b did not have relevant research datasets of their own and used those from the usage scenarios; the three others used their own data. Supplemental Table S2 provides detailed participant characteristics.

8.5.2 *Procedure*

We requested they send us one or two of their own research datasets a few days before the session and added them to ADView ourselves to ensure they were ready for immediate use; in some cases we undertook minimal pre-processing effort such as mapping machine-friendly sequence labels into meaningful human-readable names. In the session, we first gave a 20-minute demonstration of the two usage scenario datasets to illustrate the system's functionality, and also explained the exact details of how their individual datasets were pre-processed. Next, the participants used ADView on their prepared datasets for roughly 50 minutes,

where they were asked to think aloud while working. Our primary goal with this study was to assess the utility of the system rather than to improve the software usability, so we encouraged them to immediately ask us any questions about the meaning of the visual encodings and how to perform certain operations, stepped in if we saw a clear misuse of the tool or implementation bugs, discussed possible follow-up actions if they were stuck, and asked questions if we were not able to understand what the participants were doing. In the final 20 minutes we asked a set of open-ended questions, documented in Supplemental Sec.S4.2. We recorded audio and video for the entire session, and took notes manually.

8.5.3 Case study 1 findings

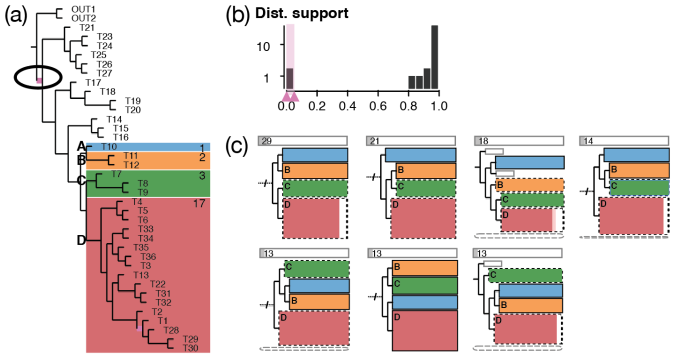


Fig. 12. Detail views of P1's findings. (a) Reference tree with four focal subtrees selected to investigate the position of T10 (blue, label A) and a branch highlighted in pink (circled). (b) Support distribution histogram with very low values selected, driving the pink highlight in the reference view. (c) Cluster ADs showing the top-frequency clusters.

P1 compared a species tree against 260 gene trees of plants. We obtained permission to show this dataset with anonymized organism names to protect unpublished data.

Sanity check on support values: upon loading the dataset, she noticed branches with extremely low support values according to the reference tree attribute distribution, shown in Fig. 12(b). She selected the leftmost bar, which highlighted one branch in thick pink in the reference tree (T2), shown in Fig. 12(a). Once she hovered on that branch, which contains all in-group taxa, the branch detail panel shows that both its support and length are zero. She clicked on this branch, and the *Tree Distribution* view showed that all trees agree with the reference tree, which seemed to conflict with the zero support value (T5). After checking a few gene trees in detail using the *Pairwise Comparison* view (T3), she figured out that the zero-support in-group branch was generated by the rooting process, and dismissed it as not relevant to her research questions.

Position of T10 She then highlighted branches with relatively low support values (0.8 to 0.9) in the same fashion (T2), and discovered a specific taxon T10 was placed differently across the gene trees. To understand the alternative positions of T10 within a bigger monophyletic group, she broke it down into several subtrees, visible as the blue, orange, green and red groups (T4) in Fig. 12(a). The first 4 of the 56 cluster ADs are shown in Fig. 12(c). The majority of the remaining clusters contained only one AD, which suggested the relationships between the focal subtrees were badly disputed. To investigate the reasons for the disagreement, she looked at the consensus trees of the clusters and pairwise compared several individual ADs to each other.

8.5.4 Case study 2 findings

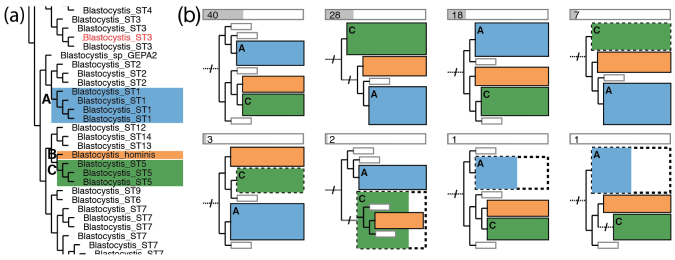


Fig. 13. Cropped figures of P2's findings: (a) The selected subtrees of interest (blue- BLASTOCYSTIS STRAIN 1, orange - BLASTOCYSTIS HOMINIS, green - BLASTOCYSTIS STRAIN 5); (b) different positions of orange taxon across bootstrapped trees.

P2 compared a consensus tree (used for the reference tree) against 100 bootstrapped trees of parasites, and approved our usage of this dataset with unsanitized labels.

Sanity check on known strains: to familiarize himself with the interface and interaction, he selected a few monophyletic groups of different parasite strains (T1). He was able to quickly confirm his knowledge of which two strains are closer than others, and the general branching pattern of strains 1 to 5, by selecting them in the reference tree and then checking the cluster ADs (T4).

Unknown outlier (orange): he identified an interesting species, BLASTOCYSTIS HOMINIS, shown as the orange group in Fig. 13(b). He knew that it usually lives in pigs, but he saw that here it was placed within the group mostly related to humans (T4). He knew that the orange group branched with the green one because he had seen it in the reference consensus tree on the left, but he had not realized that there is uncertainty about placement of the orange group until he saw the cluster ADs. Fig. 13(b) shows that only 40 trees out of 100 have the same branching pattern as the reference tree, whereas there are 29 trees suggesting that the orange is closer to the blue and that the green branched before the other two. He then constructed a sub-collection of each cluster and pairwise compared some trees one by one within each sub-collection against the consensus tree for more detail (T3).

8.5.5 Feedback

There was general agreement from all participants that many aspects of the design were successful. The task and data abstractions matched their biological intuitions, as did the results of the corresponding branches algorithm. The overall AD design showed many trees in a way that was considered quite intuitive, despite being a completely new visual representation to all participants. They all could see themselves using ADView in upcoming research projects to analyze the trees generated by a reconstruction pipeline.

Participants also noted limitations of ADView. Most thought it was complicated to learn, and they would need more experience using it to fully exploit its power. One minor point of confusion came at startup time when no

subtrees are selected, where the ADs consist of only a single blank rectangle. While most participants agreed that the *Tree Distribution* view was useful for exploring agreement and conflicts of a selected subtree, it was sufficiently unusual that two participants did not fully understand its meaning until a discussion in the question period at the end. One participant observed its connection to an existing visualization that shows the distribution of alternative taxa groupings with a small pie chart next to a tree branch [42, Fig. 2]; she noted that we had "unrolled" a pie into a horizontal bar.

Although the main goal of this study was not usability testing, we did make design refinements after each session to address awkward interactions. Most were minor interaction improvements, but a notable late-stage addition was to automatically compute appropriate sizes for the ADs and pass that request to the layout algorithm, rather then rely on the user to explicitly set sizes through a manual slider.

9 Discussion

ADView can be used by researchers to speed up the overall process of phylogenetic analysis at multiple stages in a project, especially exploratory analysis right after obtaining the trees and also to generate figures to communicate findings near the end of projects. We envision a workflow where researchers would typically follow up their initial explorations with more rigorous statistical tools. It is difficult to document the amount of speedup quantitatively, since biology papers do not typically provide any explicit estimates of how long their original analysis process required, as with our usage scenario source [49]. However, our biologist coauthor has extensive experience in analyzing this kind of data. His assessment is that ADView provides substantial speedups in assessing the quality of data, process debugging and other sanity checking, identifying interesting areas, gathering initial evidence, and generating hypotheses.

ADView can handle datasets of hundreds of trees with about 100 leaf nodes per tree from an algorithmic point of view. The de facto limit on the number of leaf nodes is primarily due to the design target of fitting the detailed reference tree dendrogram on a single screen without scrolling. There is no hard limit on the number of trees for our proposed method, but our current implementation on the browser can only deal with roughly 500 trees at interactive frame rates before the number of SVG elements would slow down the rendering process. However, for larger datasets, the user can filter out many trees and analyze smaller subsets of dozens to hundreds of trees, or simply close the *Individual AD* view and achieve fast response time by relying solely on the *Cluster AD* view. Addressing these algorithmic bottlenecks in future work would allow studies to determine the perceptual scalability limits of our approach.

10 Conclusion and Future Work

In this paper, we provide task and data abstractions for the problem of one-to-many tree comparison in the domain of phylogeny. We propose a new technique, the aggregated dendrogram, to summarize the topological relationships between focal subtrees via a simple, intuitive, and effective representation. We provide an algorithm that adapts to available screen space. We present the design and implemention of ADView, which provides insight into a tree collection of hundreds of trees at five levels of detail: the full collection, a subset of trees, single tree, subtree, and individual tree branches. We validate our approach with empirical benchmarks, direct comparison to the most similar previous system [5], usage scenarios, and an interview study conducted with five domain experts including case studies with their findings from using the system.

Extending ADView to handle duplicate leaf nodes, which arise from common evolutionary events such as gene duplication, would be an excellent and challenging direction for future work. It would also be interesting to see what aspects of ADView are useful in general tree comparison problems outside of the phylogeny domain.

Acknowledgments

SHZ was supported by NSERC grant RGPIN-2016-03711 awarded to Sarah P. Otto. ZPL was supported by NSERC grant RGPIN-2014-06309. We thank biologists Sarah P. Otto, Sean Graham, and Wayne Maddison for their advice. We appreciate comments on paper drafts from Anamaria Crisan, Kim Dextras-Romagnino, Madison Elliott, and Michael Oppermann.

References

[1] E. M. Lemmon and A. R. Lemmon, "High-throughput genomic data in systematics and phylogenetics," *Annual Review of Ecology, Evolution, and Systematics*, vol. 44, no. 1, pp. 99–121, 2013.

[2] M. Graham and J. Kennedy, "A survey of multiple tree visualisation," *Information Visualization*, vol. 9, no. 4, pp. 235–252, 2010.

[3] T. Munzner, F. Guimbretière, S. Tasiran, L. Zhang, and Y. Zhou, "TreeJuxtaposer: Scalable tree comparison using focus+context with guaranteed visibility," *ACM Trans. Graph.*, vol. 22, no. 3, pp. 453–462, 2003.

[4] D. M. Hillis, T. A. Heath, and K. St John, "Analysis and visualization of tree space," *Systematic Biology*, vol. 54, no. 3, pp. 471–482, 2005.

[5] S. Bremm, T. Von Landesberger, M. Heß, T. Schreck, P. Weil, and K. Hamacher, "Interactive visual comparison of multiple trees," in *IEEE Conf. Visual Analytics Science and Technology (VAST)*, 2011, pp. 31–40.

[6] J. Zhao, F. Chevalier, C. Collins, and R. Balakrishnan, "Facilitating discourse analysis with interactive visualization," *IEEE Trans. Visualization & Computer Graphics*, vol. 18, no. 12, pp. 2639–2648, 2012.

[7] H.-J. Schulz, "Treevis.net: A tree visualization reference," *IEEE Computer Graphics and Applications*, vol. 31, no. 6, pp. 11–15, 2011.

[8] M. Gleicher, "Considerations for visualizing comparison," *IEEE Trans. Visualization & Computer Graphics*, vol. 24, no. 1, pp. 413–423, 2018.

[9] M. J. McGuffin and J.-M. Robert, "Quantifying the space-efficiency of 2D graphical representations of trees," *Information Visualization*, vol. 9, no. 2, pp. 115–140, 2010.

[10] D. Auber, "Using strahler numbers for real time visual exploration of huge graphs," in *Proc. Intl. Conf. Computer Vision and Graphics*, vol. 1, 2002, p. 3.

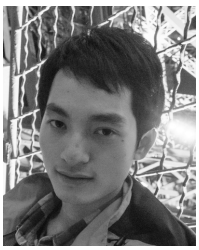
[11] M. Mehta, J. Rissanen, and R. Agrawal, "MDL-based decision tree pruning." in *Intl. Conf. Knowledge Discovery in Databases and Data Mining (KDD)*, vol. 21, no. 2, 1995, pp. 216–221.

[12] J. Heer and S. K. Card, "DOITrees revisited: scalable, space-constrained visualization of hierarchical data," in *Proc. Intl. Working Conf. Advanced Visual Interfaces (AVI)*. ACM, 2004, pp. 421–424.

[13] A. J. Pretorius, I. A. Khan, and R. J. Errington, "Cell lineage visualisation," in *Computer Graphics Forum*, vol. 34, no. 3, 2015, pp. 21–30.

[14] A. Rambaut, "FigTree," http://tree.bio.ed.ac.uk/software/figtree/.

[15] I. Letunic and P. Bork, "Interactive tree of life (iTOL) v3: an online tool for the display and annotation of phylogenetic and other trees," *Nucleic acids research*, vol. 44, no. W1, pp. W242–W245, 2016.
[16] D. H. Huson and C. Scornavacca, "Dendroscope 3: an interactive tool for rooted phylogenetic trees and networks," *Systematic biology*, vol. 61, no. 6, pp. 1061–1067, 2012.
[17] J. Sukumaran and M. T. Holder, "DendroPy: a python library for phylogenetic computing," *Bioinformatics*, vol. 26, no. 12, pp. 1569–1571, 2010.
[18] E. Paradis, J. Claude, and K. Strimmer, "APE: analyses of phylogenetics and evolution in R language," *Bioinformatics*, vol. 20, pp. 289–290, 2004.
[19] C. Plaisant, J. D. Fekete, and G. Grinstein, "Promoting insight-based evaluation of visualizations: From contest to benchmark repository," *IEEE Trans. Visualization & Computer Graphics*, vol. 14, no. 1, pp. 120–134, 2008.
[20] O. Robinson, D. Dylus, and C. Dessimoz, "Phylo.io: interactive viewing and comparison of large phylogenetic trees on the web," *Molecular biology and evolution*, vol. 33, no. 8, pp. 2163–2166, 2016.
[21] B. Lee, G. G. Robertson, M. Czerwinski, and C. S. Parr, "CandidTree: visualizing structural uncertainty in similar hierarchies," *Information Visualization*, vol. 6, no. 3, pp. 233–246, 2007.
[22] F. Beck, F.-J. Wiszniewsky, M. Burch, S. Diehl, and D. Weiskopf, "Asymmetric visual hierarchy comparison with nested icicle plots." in *Joint Proc. Intl. Workshop on Euler Diagrams & Graph Visualization in Practice*, 2014, pp. 53–62.
[23] D. Holten and J. J. Van Wijk, "Visual comparison of hierarchically organized data," in *Computer Graphics Forum*, 2008, pp. 759–766.
[24] B. Venkatachalam, J. Apple, K. St John, and D. Gusfield, "Untangling tanglegrams: Comparing trees by their drawings," *IEEE/ACM Trans. Computational Biology and Bioinformatics (TCBB)*, vol. 7, no. 4, pp. 588–597, 2010.
[25] T. Dwyer and F. Schreiber, "Optimal leaf ordering for two and a half dimensional phylogenetic tree visualisation," in *Proc. Australasian Symp. Information Visualization*, vol. 35, 2004, pp. 109–115.
[26] D. Archambault, "Structural differences between two graphs through hierarchies," in *Proc. Graphics Interface*. Canadian Information Processing Society, 2009, pp. 87–94.
[27] K. Strimmer and A. Von Haeseler, "Likelihood-mapping: a simple method to visualize phylogenetic content of a sequence alignment," *Proc. National Academy of Sciences*, vol. 94, no. 13, pp. 6815–6819, 1997.
[28] J. Fangerau, B. Höckendorf, B. Rieck, C. Heine, J. Wittbrodt, and H. Leitte, "Interactive similarity analysis and error detection in large tree collections," in *Visualization in Medicine and Life Sciences III*. Springer, 2016, pp. 287–307.
[29] M. Hess, S. Bremm, S. Weissgraeber, K. Hamacher, M. Goesele, J. Wiemeyer, and T. von Landesberger, "Visual exploration of parameter influence on phylogenetic trees," *IEEE computer graphics and applications*, vol. 34, no. 2, pp. 48–56, 2014.
[30] S. van den Elzen, D. Holten, J. Blaas, and J. J. van Wijk, "Reducing snapshots to points: A visual analytics approach to dynamic network exploration," *IEEE Trans. Visualization & Computer Graphics*, vol. 22, no. 1, pp. 1–10, 2016.
[31] M. Graham and J. Kennedy, "Multiform views of multiple trees," in *Intl. Conf. Information Visualisation*, 2008, pp. 252–257.
[32] M. Glueck, P. Hamilton, F. Chevalier, S. Breslav, A. Khan, D. Wigdor, and M. Brudno, "PhenoBlocks: Phenotype comparison visualizations," *IEEE Trans. Visualization & Computer Graphics*, vol. 22, no. 1, pp. 101–110, 2016.
[33] J. Guerra-Gomez, M. L. Pack, C. Plaisant, and B. Shneiderman, "Visualizing change over time using dynamic hierarchies: TreeVersity2 and the StemView," *IEEE Trans. Visualization & Computer Graphics*, vol. 19, no. 12, pp. 2566–2575, 2013.
[34] C. Vehlow, F. Beck, and D. Weiskopf, "Visualizing dynamic hierarchies in graph sequences," *IEEE Trans. Visualization & Computer Graphics*, vol. 22, no. 10, pp. 2343–2357, 2016.
[35] B. Bach, N. Henry-Riche, T. Dwyer, T. Madhyastha, J.-D. Fekete, and T. Grabowski, "Small MultiPiles: Piling time to explore temporal patterns in dynamic networks," in *Computer Graphics Forum*, vol. 34, no. 3, 2015, pp. 31–40.
[36] D. Bryant, "A classification of consensus methods for phylogenetics," *DIMACS series in discrete mathematics and theoretical computer science*, vol. 61, pp. 163–184, 2003.
[37] D. Koop, J. Freire, and C. T. Silva, "Visual summaries for graph collections," in *IEEE Symp. Pacific Visualization (PacificVis)*. IEEE, 2013, pp. 57–64.
[38] R. Bouckaert and J. Heled, "DensiTree 2: seeing trees through the forest," *bioRxiv*, p. 012401, 2014.
[39] B. Lee, C. Plaisant, C. S. Parr, J.-D. Fekete, and N. Henry, "Task taxonomy for graph visualization," in *Proc. AVI workshop on BEyond time and errors: novel evaluation methods for information visualization*. ACM, 2006, pp. 1–5.
[40] J. Pretorius, H. C. Purchase, and J. T. Stasko, "Tasks for multivariate network analysis," in *Multivariate network visualization*. Springer, 2014, pp. 77–95.
[41] N. Kerracher, J. Kennedy, and K. Chalmers, "A task taxonomy for temporal graph visualisation," *IEEE Trans. Visualization & Computer Graphics*, vol. 21, no. 10, pp. 1160–1172, 2015.
[42] S. A. Smith, M. J. Moore, J. W. Brown, and Y. Yang, "Analysis of phylogenomic datasets reveals conflict, concordance, and gene duplications with examples from animals and plants," *BMC Evolutionary Biology*, vol. 15, no. 1, p. 150, 2015.
[43] N. Kerracher and J. Kennedy, "Constructing and evaluating visualisation task classifications: Process and considerations," *Computer Graphics Forum*, vol. 36, no. 3, pp. 47–59, 2017.
[44] T. Munzner, *Visualization Analysis & Design*. CRC Press, AK Peters Visualization Series, 2014.
[45] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *Journ. Machine Learning Research*, vol. 9, no. 2579-2605, p. 85, 2008.
[46] Z. Qu and J. Hullman, "Keeping multiple views consistent: Constraints, validations, and exceptions in visualization authoring," *IEEE Trans. Visualization & Computer Graphics*, vol. 24, no. 1, pp. 468–477, 2018.
[47] "Newick tree format," http://evolution.genetics.washington.edu/phylip/newicktree.html.
[48] D. Robinson and L. Foulds, "Comparison of phylogenetic trees," *Mathematical Biosciences*, vol. 53, no. 1, pp. 131 – 147, 1981.
[49] N. J. Wickett *et al.*, "Phylotranscriptomic analysis of the origin and early diversification of land plants," *Proc. National Academy of Sciences*, vol. 111, no. 45, pp. E4859–E4868, 2014.



Zipeng Liu is a PhD student in the Department of Computer Science at the University of British Columbia, in the InfoVis group. He received his BS degree from Peking University in 2015. His research involves devising novel visualization technique and developing applications for biology.

Shing Hei Zhan is a PhD candidate in the Bioinformatics program at the University of British Columbia. His research involves the development of phylogenomic methodologies and the large-scale analysis of trait evolution of various organisms, in particular plants and algae.

Tamara Munzner is a professor at the University of British Columbia, and holds a PhD from Stanford from 2000. She has co-chaired InfoVis and EuroVis, her book *Visualization Analysis and Design* appeared in 2014, and she received the IEEE VGTC Visualization Technical Achievement Award in 2015. She has worked on visualization projects in a broad range of application domains from genomics to journalism.