Computational Geometry: Theory and Applications 116 (2024) 102031

Contents lists available at ScienceDirect

Computational Geometry: Theory and Applications

journal homepage: www.elsevier.com/locate/comgeo

From trees to barcodes and back again II: Combinatorial and probabilistic aspects of a topological inverse problem

Justin Curry, Jordan DeSha, Adélie Garin *, Kathryn Hess, Lida Kanari, Brendan Mallery

ARTICLE INFO

Article history:
Received 23 July 2021
Received in revised form 13 January 2023
Accepted 9 June 2023
Available online 22 June 2023

Keywords:
Topological data analysis
Inverse problem
Persistent homology

ABSTRACT

In this paper we consider two aspects of the inverse problem of how to construct merge trees realizing a given barcode. Much of our investigation exploits a recently discovered connection between the symmetric group and barcodes in general position, based on the simple observation that death order is a permutation of birth order. We show how to lift this combinatorial characterization of barcodes to an analogous combinatorialization of merge trees. As result of this study, we provide the first clear combinatorial distinction between the space of phylogenetic trees (as defined by Billera, Holmes and Vogtmann) and the space of merge trees: generic phylogenetic trees on $n+1$ leaf nodes fall into $(2n-1)!!$ distinct equivalence classes, but the analogous number for merge trees is equal to the number of maximal chains in the lattice of partitions, i.e., $(n+1)!n!2^{-n}$. The second aspect of our study is the derivation of precise formulas for the distribution of tree realization numbers (the number of merge trees realizing a given barcode) when we assume that barcodes are sampled using a uniform distribution on the symmetric group. We are able to characterize some of the higher moments of this distribution, thanks in part to a reformulation of our distribution in terms of Dirichlet convolution. This characterization provides a type of null hypothesis, apparently different from the distributions observed in real neuron data, which opens the door to doing more precise statistics and science.

Contents

* Corresponding author.
E-mail address: adelie.garin@gmail.com (A. Garin).

https://doi.org/10.1016/j.comgeo.2023.102031

1. Introduction

Trees have a nearly universal presence as a structure for organizing relationships between objects. From hierarchical arrangements that are useful in the classification of species, to more immediate geometric applications in modeling neuron morphology [15–17], trees have proved to be an indispensable tool. However, as is natural for such a universal concept, subtle variations introduce important differences that are not always commented on. In this paper, we are interested in the comparison of the notion of *merge trees*, which is an important tool in topological data analysis (TDA), and that of *metric phylogenetic trees*, which has gained a tremendous traction since its formalization by Billera, Holmes and Vogtmann [1], along with their combinatorial variants.

Our interest in delineating these objects comes in part from the fact that both merge trees and metric phylogenetic trees have associated *barcodes*, which are topological invariants obtained from the persistent homology of a filtered space.

Since their introduction, barcodes or *persistence diagrams* have become the standard topological summary used in TDA. Like all summaries, barcodes forget information about the space they are computed from. Thus, even when restricting to a specific set of topological spaces like trees, one may find that many different shapes give rise to the same barcode. Quantifying this failure of injectivity into a summary space is the realm of *topological inverse problems*. Understanding such problems is crucial for comparing different representations of objects arising in both pure mathematics and in data science.

1.1. High-level overview and motivation

In this paper we consider two aspects of the inverse problem of constructing merge trees realizing a given barcode, motivated by recent work in neuroscience. In particular, the tools developed in [15,16] have proven useful for the study of neuron morphologies [17], which can be modeled by rooted trees, i.e., acyclic binary graphs with a distinguished vertex called the root (which corresponds to the neuron's soma), embedded in $\mathbb{R}^3$. In the terminology of this paper, this structure is most faithfully represented by merge trees when neurons are equipped with the intrinsic path distance.

In [16] the authors introduced the Topological Morphology Descriptor (TMD), an algorithm that returns a barcode from a tree, keeping track of the lengths of each branch with respect to a given filtration, but forgetting the adjacency relations between the branches. In this article we expand this investigation and systematically study the inverse problem from a combinatorial point of view. We hope that understanding this relation will provide insight into the complex structures of neurons; see Fig. 1 for a schematic.

The general approach to the merge tree-to-barcode inverse problem is as follows. Any barcode can be realized by finitely many trees, the number of which is called the tree realization number (TRN) or simply the realization number of the barcode. As observed in [5] and [18], the realization number of a barcode in general position can be computed by certain containment relations between its bars, viewed as intervals on the real line. One of the crucial observations of [18] is that these containment relations partition the set of barcodes (on n bars) into equivalence classes, indexed by permutations in S_n, the symmetric group on n letters.

The representation of a barcode by a permutation not only gives a formula for the tree realization number (Lemma 3.4), but also opens the door to deeper connections between inverse problems in TDA, group theory, and combinatorics.

Besides quantifying the relative "descriptive power" of different summaries, in [18] it was shown that the realization number could be used as a statistic to distinguish distributions of trees. Fig. 2 shows (log) realization numbers computed from different tree distributions, obtained by computing the realization number either from actual trees, such as neurons, or by randomly generating barcodes with specific properties. The datasets used were *(i)* real neurons (basal and apical dendrites, drawn in red and purple), *(ii)* random barcodes where the birth b_i is picked, then the death d_i is chosen to be larger than b_i, and *(iii)* random barcodes with separated births and deaths so that the induced distribution on the symmetric group is uniform (see Section 4.1). The results are striking: barcodes computed from neurons exhibit a very different distribution than barcodes with uniformly drawn permutation type; see Fig. 2 for a graphical comparison.

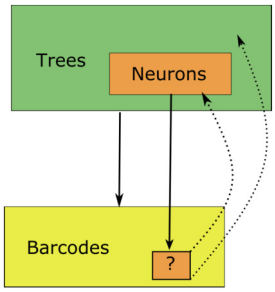


Fig. 1. Motivation for understanding the pre-image of a barcode: Given a barcode computed from a neuron, what do all of its pre-images look like?

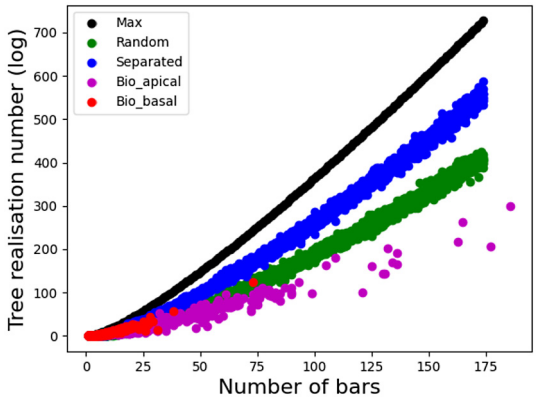


Fig. 2. The log of the tree-realization number for barcodes with varying numbers of bars for TMD of basal dendrites (red), apical dendrites (purple) in comparison with "random" barcodes as defined in Section 4.1 (green), barcodes with separated births and deaths such that the distribution induced on the symmetric group is uniform (blue, see section 4.1 and Proposition 4.4), and the maximum tree-realization number ($n!$ for $n+1$ bars) (black). (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

In this paper, we study the realization numbers computed from barcodes with uniform permutation type (i.e., drawn from the uniform distribution on the symmetric group). We view this as essential for the realization number to be used for applications, as it establishes a fundamental null hypothesis for the invariant. Our tools are mainly combinatorial, leading us to discover unexpected connections between the inverse problem and other classical combinatorial objects. One of our main theorems (Theorem 3.21) casts the classic result of Erdős that counts the number of maximal chains in the lattice of set partitions in a new, merge-tree light. It was this result that not only permitted an easy calculation of the expected tree realization number, but also further established the fundamental differences between combinatorial classes of merge trees and phylogenetic trees, which is detailed in the appendix. We now provide a more detailed overview of the paper.

1.2. Detailed overview

After the introduction, we start in earnest by reviewing the basic properties of trees and barcodes in Section 2. The basic graph-theoretic notion of a tree is reviewed in Definition 2.1, as are the notions of labeling and isomorphism. These notions are necessary to review because merge trees (Definition 2.3) come equipped with a height function, which can induce a labeling, but not one that is compatible with the notion of equivalence used for phylogenetic trees. These distinctions are made precise in the Appendix, where these concepts are reviewed. We mention that Proposition A.3 provides a more carefully stated distinction between the notions of phylogenetic tree (BHV) space, labeled merge tree space, and merge tree space for the interested reader. However, the primary concept of Section 2 is a new combinatorial notion of a merge tree. The pertinence of this combinatorial notion becomes evident after we introduce barcodes in Section 2.2, which allows us to review the inverse problem for merge trees in Section 2.3, where the combinatorial (permutation) type of the barcode is all that matters (see Section 2.4).

Section 3 marks the true beginning of this paper's contribution to the literature. In Section 3.1, we formalize the observation of [18] that the tree realization number (TRN) is a function of the symmetric group, by expressing the TRN in terms of the left-inversion vector associated to a permutation. We take a minor detour in Section 3.2 to observe that the combinatorial equivalence class of each barcode is convex (Lemma 3.6), which is of use later when we choose certain standard forms for barcodes (Definition 3.9) and merge trees (Definition 3.17). We continue the algebraic analysis of the TRN in Section 3.3, where we prove that when the symmetric group is equipped with a certain partial order (Definition 3.11), the TRN is an order-preserving map. After proving that every pair of combinatorially equivalent merge trees can be connected by a line of merge trees (Lemma 3.16), we show that the sum of the tree realization numbers is equal to the total

number of combinatorial types of merge trees in Lemma 3.19. Theorem 3.21 in turn states that this number is equal to the number of maximal chains in the lattice of partitions (Definition 3.20), which is $(n+1)!n!2^{-n}$. This result provides a stark combinatorial contrast with the well-known fact that there are $(2n-1)!!$ types of labeled binary trees on $n+1$ nodes [11]. Section A.1 explores this contrast in greater depth by making quantitative the observation that whereas merge trees fiber over the symmetric group in a nice way, phylogenetic trees do not.

Section 4 finally delivers closed-form formulas for some of the trend lines in Fig. 2. We cover briefly two methods to generate random barcodes in Section 4.1, before characterizing the distribution of tree realization numbers (when sampled uniformly on the symmetric group) in terms of Dirichlet convolution in Theorem 4.1. The paper concludes with Proposition 4.8, which uses the left-inversion vector representation of the TRN to give a closed formula for the expected log realization number.

1.3. Related work

This paper touches on many classical concepts related to trees and combinatorics, so providing a complete list of related work is impossible. However, the literature on inverse problems for TDA can be reviewed briefly here.

The concept of a geometric realization of a persistence module was considered in [20] in order to prove a universality result for the interleaving distance. In [13] the authors initiated an algorithmic study of how to find a point cloud that realizes a given persistence diagram. While these articles are concerned with finding single realizations of persistent signatures, the present article focuses on the study of the entire pre-image of the persistent homology pipeline.

In the same vein, there is [5], which focused on the setting of functions on the interval and their associated merge trees. Some of the results there were independently rediscovered and extended in [18], which inspired the present collaboration. Both [8] and [21] are more recent articles that investigate the fiber of the persistence map in settings that are different from ours.

We note that the study of the (non-) injectivity of certain topological transforms is also an aspect of topological inverse problems, see [24,14,7,22,26] for a sampling of these articles and [25] for a recent survey. Better understanding the precise failure of injectivity of certain TDA invariants led to the development of enriched topological summaries (ETS) that remediate these failures, opening a promising line of research; see [3] and [6] for some examples of these ETS.

Section 3.3 of this paper explores the relationship between the Bruhat order on the symmetric group and barcode equivalence classes. A similar connection was observed in [27] in a different context. Later work [2] provides a full description of the space of barcodes in terms of a stratification using Coxeter complexes, objects which are strongly related to the symmetric group.

2. Background on trees and barcodes

In this section, we assume basic familiarity with persistent homology in degree 0, even though it is not necessary to understand persistence for most of these definitions. For a more algorithmic review of the topic in the case of trees, see [18]. We begin by reviewing the necessary background and combinatorial results from [5] and [18]. Most of this section reviews prior work, though after an initial reading the reader may go to Proposition A.3 to see a novel comparison of merge trees and phylogenetic trees, which motivates much of the paper.

2.1. Trees, merge trees and combinatorial equivalence

There are many notions of trees in mathematics and the sciences. We review a few of these here and explain their differences. Note that in this paper, we only consider binary trees. Up to a small deformation, any non-binary tree can be made binary. Moreover, in real life data, tree structures such as neurons are almost always binary, because the probability to have 2 separate branches growing exactly at the same spot is 0. We start with the simplest definition, that of a combinatorial tree.

Definition 2.1. A *combinatorial tree* T is a connected, acyclic, binary graph. It is *finite* if the number of vertices is finite. A *rooted tree* is a combinatorial tree with a distinguished vertex of degree 1 called the *root*. Non-root vertices of degree 1 are called *leaves*.

A *labeling* of a combinatorial tree T is a bijective map from its set of vertices $V(T)$ to a set S of labels. A labeling is *ordered* if S is a subset of the natural numbers $\mathbb{N}$. An ordered labeling of a tree with n vertices gives rise to an $n \times n$ *adjacency matrix*, of which the (i,j)-coefficient is 1 if there is an edge between the vertices labeled i and j and is 0 otherwise.

Two combinatorial trees T and T' are *isomorphic* if there is a bijective map $T \to T'$ that sends vertices to vertices in an adjacency-preserving way: if two vertices in T are connected by an edge, then so are their images. Equivalently, T and T' are isomorphic if there exist ordered labellings of both with respect to which their adjacency matrices are identical.

In this paper, we assume all trees are finite. Moreover, we assume that there are no vertices of degree 2, that is, each vertex is either a *bifurcation* or *branching* point, i.e., a vertex of degree 3, or a *termination*, i.e., a vertex of degree 1, such as the leaf nodes or the root.

When rooted trees are considered, there is a natural way to induce an orientation on the edges of the tree: for each vertex v, there is a unique path from v to the root r. Every edge of the tree is oriented from the vertex further from r to the closer one (with respect to the graph path distance). A vertex v of T is a *parent* of a vertex w if there is a directed edge from w to v; the vertex w is then a *child* of v. If there is a sequence of directed edges from w to v then v is an *ancestor* of w. Each vertex of T has a unique parent, except for the root r, which has no parent at all.

Remark 2.2. The *path-metric* between two nodes of a graph, sometimes called the *hop-metric*, is the number of edges on the shortest path between the two nodes. The notion of v being an ancestor of w is equivalent to v being on the path between w and the root r, hence the distance between v and r in the path metric is shorter than the one between w and r.

Note that a finite combinatorial tree T is fully specified by its set of vertices, equipped with the partial order specified by the "is a parent of" relation.

The language of "parents" and "children" obviously comes from studying ancestral relations for people (as in family trees) and species (as in phylogenetic trees). There are also situations where the parent-child relation is determined in part by a notion of "height," which is how merge trees are defined.

Definition 2.3. A *merge tree* is a rooted combinatorial tree T, together with a function on the vertices $h : V(T) \longrightarrow \mathbb{R} \cup \{\infty\}$, called a *height function*, that satisfies two properties.

(1) If v is the parent of w, then $h(v) \geq h(w)$.
(2) If r is the root node, then $h(r) = \infty$.

Two merge trees (T,h) and (T',h') are *isomorphic* if there is a graph isomorphism $\varphi : T \to T'$ that preserves heights, i.e., $h = h' \circ \varphi$. A *generic (or strict) merge tree* is a merge tree (T,h) such that the height function $h : V(T) \to \mathbb{R} \cup \infty$ is injective. We always assume our merge trees are generic, unless otherwise indicated.

Remark 2.4 *(Drawing conventions for merge trees).* Many authors choose to draw merge trees so that the function $h : V(T) \to \mathbb{R} \cup \infty$ resembles height when embedded in the page. This has the effect of placing the root node higher than the leaf nodes, contrary to how trees appear in nature. To honor the natural orientation and size of trees in nature, we draw our merge trees with the opposite convention, so that the root is lower than the leaves and so that $f(r) = \infty$ is represented with a finite value N.

Remark 2.5 *(Alternative definition of merge trees).* Another, perhaps more common, definition of a merge tree is that it is the Reeb graph of the epigraph of a function on a topological space X. It should be noted that such a tree is binary only under genericity conditions on f. From this point of view, the merge tree T of a real-valued function $f : X \to \mathbb{R}$ is the quotient space of the epigraph $\Gamma^+ := \{(x,t) \in X \times \mathbb{R} \mid f(x) \leq t\}$ by the equivalence relation specified by $(x,t) \sim (y,s)$ if and only if $s = t$ and x and y are in the same path component of the sublevel set filtration of f at t, i.e., $[x] = [y] \in \pi_0(f^{-1}(-\infty,t])$. Since the projection map from Γ^+ onto the second coordinate is constant on equivalence classes, this projection map factors to define the height function. Under reasonable tameness conditions, the quotient space is homeomorphic to the geometric realization of a combinatorial tree, where vertices correspond to connected components of "critical" points.

Example 2.6. A typical example of merge tree is one arising from measuring height on an embedded manifold $X \subseteq \mathbb{R}^n$. Here "height" can be thought of as the scalar product with a specified unit vector. Fig. 3 shows a simple example of a topological space and the corresponding merge tree.

There is a natural ordered labeling on the vertices of a generic merge tree (T,h), inherited from the function h, by ordering the vertices according to their h-value: the leaf node with lowest h-value is labeled 0, and the remaining nodes are labeled based thereafter on the order in which they appear. We call the labels on the leaves the *birth labels* and the ones on the internal vertices the *death labels*, for reasons that will become clear later in the paper when we review persistent homology.

We are now in a position to state the first novel definition of the paper. Recall that two graphs are isomorphic if they admit ordered labellings making their adjacency matrices the same. A merge tree includes the additional data of heights of each node. By focusing separately on the order of births and the order of deaths, along with adjacency data, we have a more flexible notion of a merge tree.

Definition 2.7. Two generic merge trees (T,h) and (T',h') are *combinatorially equivalent* if they are isomorphic as graphs via a graph isomorphism preserving the orders of births and of deaths, respectively. In more detail, (T,h) and (T',h') are combinatorially equivalent if there exists a graph isomorphism $\varphi : T \to T'$ such that the following conditions hold.

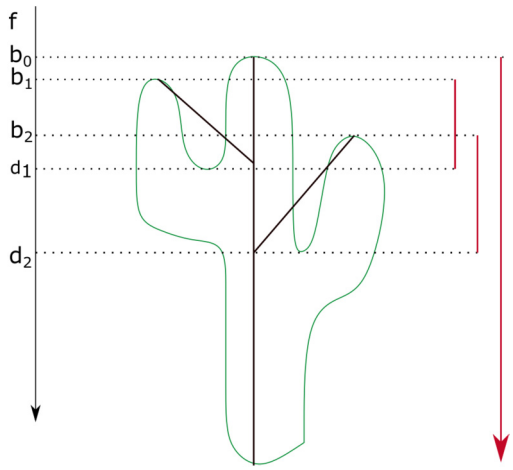


Fig. 3. A circle X is embedded in $\mathbb{R}^2$ and drawn in green to resemble a cactus with the height function f measuring distance down the page. The corresponding merge tree (Definition 2.3) is drawn in black. The barcode of the persistence module in degree 0 (Definition 2.11) associated to (X, f) is shown in red on the right.

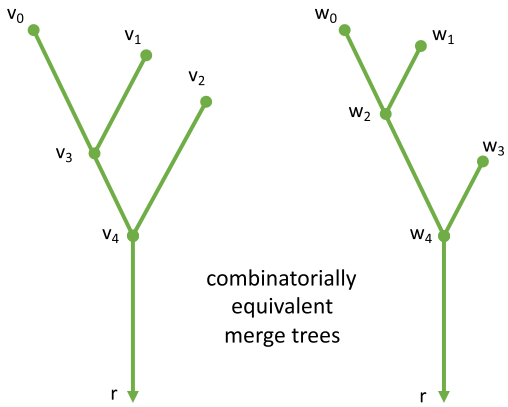


Fig. 4. Two combinatorially equivalent merge trees are shown. Notice that the total order of the vertices is not preserved, but the orders among leaf nodes and internal nodes are preserved separately.

(1) For every pair of leaf (birth) nodes v_i and v_j in T, if $h(v_i) < h(v_j)$, then $h'(\varphi(v_i)) < h'(\varphi(v_j))$.
(2) For every pair of internal (death) nodes v_i and v_j in T, if $h(v_i) < h(v_j)$, then $h'(\varphi(v_i)) < h'(\varphi(v_j))$.

We note that these two conditions specify two different sets for the logical quantifier and that the total order on vertices need not be preserved; see Fig. 4 for an example.

Remark 2.8 *(Combinatorial merge trees).* Note that combinatorial equivalence classes of merge trees are simply combinatorial trees equipped with an ordered labeling L_l of the leaves (a birth label) and an ordered labeling L_i of the internal nodes (a death label) such that the label $L_i(v)$ of internal node v is larger than the label $L_i(w)$ of internal node w if v is an ancestor of w. We call such a tree a *combinatorial merge tree*. A combinatorial tree has forgotten its geometry in favor of combinatorics.

Example 2.9 *(Translation invariant).* Consider two generic merge trees (T, h) and (T, h') such that $h' = h + \Delta$ for some real number Δ. We say (T', h') is a *translation* of T. A generic merge tree is combinatorially equivalent to any translation of itself. However, combinatorial equivalence detects relationships more general than translation; see Fig. 4.

Example 2.10 *(Sensitivity to generators).* Although the two merge trees in Fig. 5 are isomorphic as graphs, the only possible graph isomorphism reverses the birth order, hence these generic merge trees are not combinatorially equivalent. Notice that the homology generator of the essential class (see Section 2.2) starts with the node labeled by 0 or A on the left hand side, while on the right hand side, it starts with the 0 or B label. This is sometimes called "instability" or "sensitivity" of generators in TDA. Together with Fig. 4, these specify the three possible combinatorial equivalence classes of merge trees with three leaf nodes.

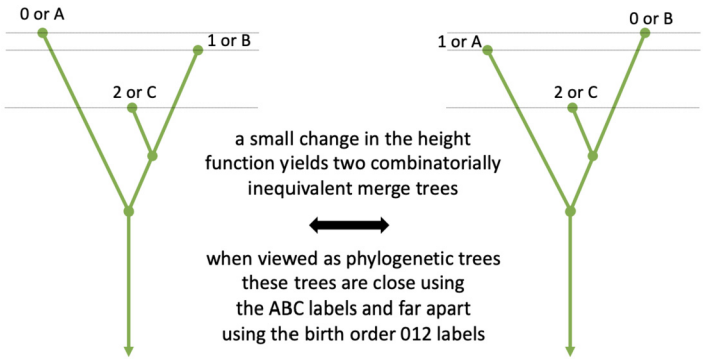


Fig. 5. Two generic merge trees that are isomorphic as graphs. When they are regarded as phylogenetic trees we fix alphabetical ('ABC') names for the leaf nodes, as if the nodes represented species that went extinct at different times. With this labeling they are considered close in the metric defined by [1]. When these are regarded as merge trees they are naturally unlabeled and are close in the interleaving distance [23], but if we use birth order ('012') to label the leaf nodes and regard them as phylogenetic trees then they are far apart; see Proposition A.3.

2.2. Barcodes

We now recall the notions of persistent homology and barcodes. For reasons of brevity, we choose to use the categorical definition of persistent homology, but the reader who would like a more algorithmic version for the case of merge trees can read [18] or the summary in Example 2.14.

Definition 2.11. A *persistence module* is a functor

$$F : (\mathbb{R}, \leq) \to Vect$$

where $(\mathbb{R}, \leq)$ is the real line with its total ordering $\leq$. Said more simply, a persistence module consists of a collection of vector spaces $F(t)$ for every real number t and a linear map $F(s \leq t) : F(s) \to F(t)$ for every pair $s \leq t$. These maps have to obey two laws: $F(s \leq s) = \mathrm{id}_{F(s)}$ and whenever $r \leq s \leq t$ is an ordered triple, the composition $F(s \leq t) \circ F(r \leq s) = F(r \leq t)$.

An *interval module* is a persistence module $\Bbbk_I$ that is rank 1 on an interval $I \subseteq \mathbb{R}$ with identity maps internal to I and 0 elsewhere.

Theorem 2.12 *(Crawley-Boevey [4]). Any pointwise finite dimensional persistence module is isomorphic to a direct sum of interval modules, and this decomposition is unique up to reordering.*

Definition 2.13. Let F be a persistence module with decomposition $F \cong \bigoplus_{j \in \mathcal{J}} \Bbbk_{I_j}^{\oplus n_j}$. The *barcode* of F is the multiset

$$B(F) = \{(I_j, n_j)\}_{j \in \mathcal{J}}.$$

In most applications, each interval I_j is of the form $[b_j, d_j)$, where b_j is the *birth* of the homological feature corresponding to I_j and d_j its *death*. We call the interval $[b_j, d_j)$ a *bar* in the barcode B.

In this paper, we represent barcodes graphically by drawing the interval between b_j and d_j for each index j. Sometimes barcodes are represented by *persistence diagrams*, i.e., sets of points in $\mathbb{R}^2$ where the x-coordinate indicates birth time and the y-coordinate death time. Note that $x \leq y$ always in this representation.

Example 2.14 *(Barcodes for merge trees and the Elder rule).* Let (T, h) be a merge tree. Regarding T as a one-dimensional simplicial complex, we can linearly interpolate the height function from the vertices to the entire tree. The *barcode of the merge tree* (T, h) is the barcode corresponding to the persistence module

$$F : (\mathbb{R}, \leq) \to Vect \qquad \text{where} \qquad F(t) = H_0\Big(h^{-1}\big((-\infty, t]\big)\Big).$$

Although the barcode of F is guaranteed to exist by virtue of Crawley-Boevey's theorem, there is a more direct way of constructing the barcode in the special case of merge trees, called the *Elder rule* [5].

The Elder rule provides a concrete way to compute the barcode of a merge tree via decomposition into branches, i.e., each bar in the barcode corresponds either to a single edge or a list of adjacent edges in the merge tree. According to the Elder rule, each leaf node marks the beginning of a bar in the barcode at the height of the leaf node. If two leaf nodes v_i and v_j such that $h(v_i) > h(v_j)$ share an ancestor at vertex k, the branch that was born "earlier" at v_j survives as it is "elder", and the branch born v_i dies, creating a bar $[h(v_i), h(v_k))$ in the barcode.

Table 1
Table summarizing the attributes of each object defined above. Labels on leaves and internal vertices of merge trees are marked by an asterisk to indicate that they are inherited from the height function. Similarly, the "labels" on barcodes (their birth and death values) are inherited from the height function on the tree.

	Combinatorial trees	Merge trees	Phylogenetic trees	Barcodes
Height function		X		
Label on leaves (births)		X*	X	X*
Label on internal vertices (deaths)		X*		X*
Adjacency	X	X	X	

Under this rule, every bar begins at a leaf node and ends at an internal node with the sole exception of the bar that is born at the leaf node with the lowest height, which is paired with infinity. However, in our figures, in keeping with Remark 2.4, the lowest leaf node will be paired with $N = f(r)$, which is the height of the root node when viewed as an embedded finite tree. A simple example is illustrated in Fig. 3.

The Elder Rule can be defined purely combinatorially for combinatorial merge trees. The output is a set of pairs of labels (the birth and death labels). Recall from Remark 2.8 that a combinatorial merge tree is a combinatorial tree T together with an ordered labeling L_l of the leaves and an ordered labeling L_i of the internal nodes, such that $L_i(v) > L_i(w)$ if v is an ancestor of w. We can define the *combinatorial Elder Rule* in the same way as the Elder Rule is defined in Example 2.14 by replacing the function f by the labellings L_l and L_i.

Definition 2.15 *(Combinatorial Elder rule).* Let (T, L_l, L_i) be a combinatorial merge tree. As in Example 2.14, each leaf node v marks the first coordinate of a pair with the label $L_l(v)$. If two leaf nodes v_i and v_j such that $L_l(v_i) < L_l(v_j)$ share an ancestor v_k, the leaf with the smallest label, v_k gets paired with v_i, creating the pair $(L_l(v_i), L_i(v_k))$. Under this rule, the leaf with the smallest label is paired with the root, which we label by ∞.

Although in general the barcode can be a true multiset, in this article we are concerned primarily with barcodes that are actually sets, leading us to formulate the following definition.

Definition 2.16. A barcode B is *strict* if is composed of one half-infinite bar $[b_0, \infty)$, and a finite number of half open bars $[b_1, d_1), \cdots, [b_n, d_n)$ such that $b_0 < \cdots < b_n$ and $d_i \neq d_j$ if $i \neq j$. We refer to the half-infinite bar as *essential*.

Example 2.17. The barcode of a generic merge tree is always strict.

We summarize the different characteristics of combinatorial trees, merge trees, phylogenetic trees, and barcodes in Table 1.

2.3. Realizations of barcodes

As described in the previous section, every merge tree has an associated barcode. It is natural to ask whether the map from merge trees to barcodes determined by the Elder rule is injective, but it is not hard to see that it is not. A somewhat more surprising result, proven independently in [5] and [18], is that the failure of injectivity of the Elder rule map can be quantified for generic barcodes. More precisely, we say that a merge tree (T, h) *realizes* a barcode B if the barcode of (T, h) is B. The *tree realization number*, $R(B)$, of a strict barcode B is the number of combinatorial trees T admitting a height function h such that (T, h) realizes B.

Proposition 2.18 *([5], [18]). Let B be a strict barcode with n finite length half-open bars $\{I_j = [b_j, d_j]\}_{j=1}^n$ and one infinite bar $I_0 = [b_0, \infty)$. The number of merge trees that realize B is*

$$R(B) = \prod_{j=1}^{n} \mu(I_j)$$

where $\mu(I_j) = |\{I_k \mid I_j \subset I_k\}|$. The value $\mu(I_j)$, called the index *of bar I_j, is the number of bars of B (including the infinite bar) that contain I_j.*

Although the proof of this theorem, by induction on the number of bars, can be found in [5] and [18], we provide a brief sketch for the sake of intuition. Start by setting $T_0 = I_0 = [b_0, \infty)$. Since the merge tree T is connected, we can recursively attach bars by death time, first to T_0 and then in the jth step to T_j to get T_{j+1}, according to the Elder rule. Each possible choice of attachment then gives a particular merge tree isomorphism class. See Fig. 7 for a graphical representation of this process.

Example 2.19. Consider the strict barcode $B = \Big\{[0,\infty),[1,8),[2,7),[3,6),[4,5)\Big\}$. According to the formula in Proposition 2.18,

$$R(B) = \prod_{j=1}^{4} \mu(I_j) = 1 \cdot 2 \cdot 3 \cdot 4 = 4!.$$

In general, if B is a strict barcode with n finite length half-open intervals such that $I_j \subset I_k$ for all $k < j$, then $R(B) = n!$.

2.4. Relations to the symmetric group

We begin by recalling the map from the set of strict barcodes with n nonessential bars to the symmetric group on n letters, which was introduced in [18].

Remark 2.20 *(Different notations for permutations).* There are several notational conventions for elements of the symmetric group. When we use square brackets or boxes, e.g., the notation [132], then we are listing the images of the ordered set $\{1,\dots,n\}$ under the map σ, e.g., for $\sigma = [132]$, one can read off that $\sigma(1) = 1$, $\sigma(2) = 3$ and $\sigma(3) = 2$. We also use cycle notation, which describes the permutation in terms of its orbits and uses parentheses; fixed points are omitted in this notation. For our example, $\sigma = [132]$ can also be written as the elementary transposition (23). See Fig. 8.

Definition 2.21. Let $B = \{[b_i,d_i)\}_{i=1}^n \cup [b_0,\infty)$ be a strict barcode such that $b_1 < \dots < b_n$. The *permutation type* σ of the barcode B is the automorphism σ of $\{1,\dots,n\}$ that maps birth order to death order. In other words, if we re-index the death times using the natural order on $\mathbb{R}$ so that $d_{i_1} < \dots < d_{i_n}$, the permutation σ is $[i_1 i_2 \dots i_n]$. In terms of the Elder rule, this *associated permutation* comes from tracking which birth is paired with which death.

Notice that the essential bar $[b_0,\infty)$ does not play a role in the permutation type, as it always contains all the other bars in a strict barcode.

The association of a permutation to each barcode defines an equivalence relation on the set of strict barcodes.

Definition 2.22. Let B and B' be two strict barcodes, each with n non-essential bars, denoted $\{[b_i,d_i)\}_{i=1}^n$ and $\{[b_i',d_i')\}_{i=1}^n$, respectively. We say B and B' are *combinatorially equivalent* if they have the same associated permutation.

We can now express the relation between barcodes and the symmetric group more concisely as follows. Let $\mathcal{B}_n$ denote the collection of strict barcodes with n finite length half-open (non-essential) bars. The map that associates to every strict barcode its permutation type defines a bijection between combinatorial equivalence classes of strict barcodes and elements of the symmetric group, i.e.,

$$\mathcal{B}_n/\sim \quad \longleftrightarrow \quad S_n.$$

Example 2.23. The space $\mathcal{B}_3/\sim$ and the corresponding elements of S_3 of the bijection given above are displayed in Fig. 6C.

Remark 2.24. As was done in Remark 2.8 for combinatorial merge trees, one can identify the combinatorial equivalence classes of barcodes with elements of the symmetric group. What will be called a *combinatorial barcode* in this paper is just the corresponding permutation in S_n.

We conclude this section by clarifying the relationship between the two notions of combinatorial equivalence that are pertinent to the tree realization problem.

Lemma 2.25. *If T and T' are combinatorially equivalent merge trees, then their corresponding barcodes B and B' are combinatorially equivalent as well.*

Proof. Since tree isomorphisms as defined in Definition 2.3 preserve both birth and death orders, we need to check only that if the Elder rule pairs the i-th birth node with the j-th death node in T, then the same holds for T'. This is obvious, however, because the unique sequence of edges connecting a pair of nodes in T must be sent to the same sequence of edges connecting these nodes in T', since φ is a graph isomorphism and therefore preserves adjacency relations. □

The following corollary follows directly.

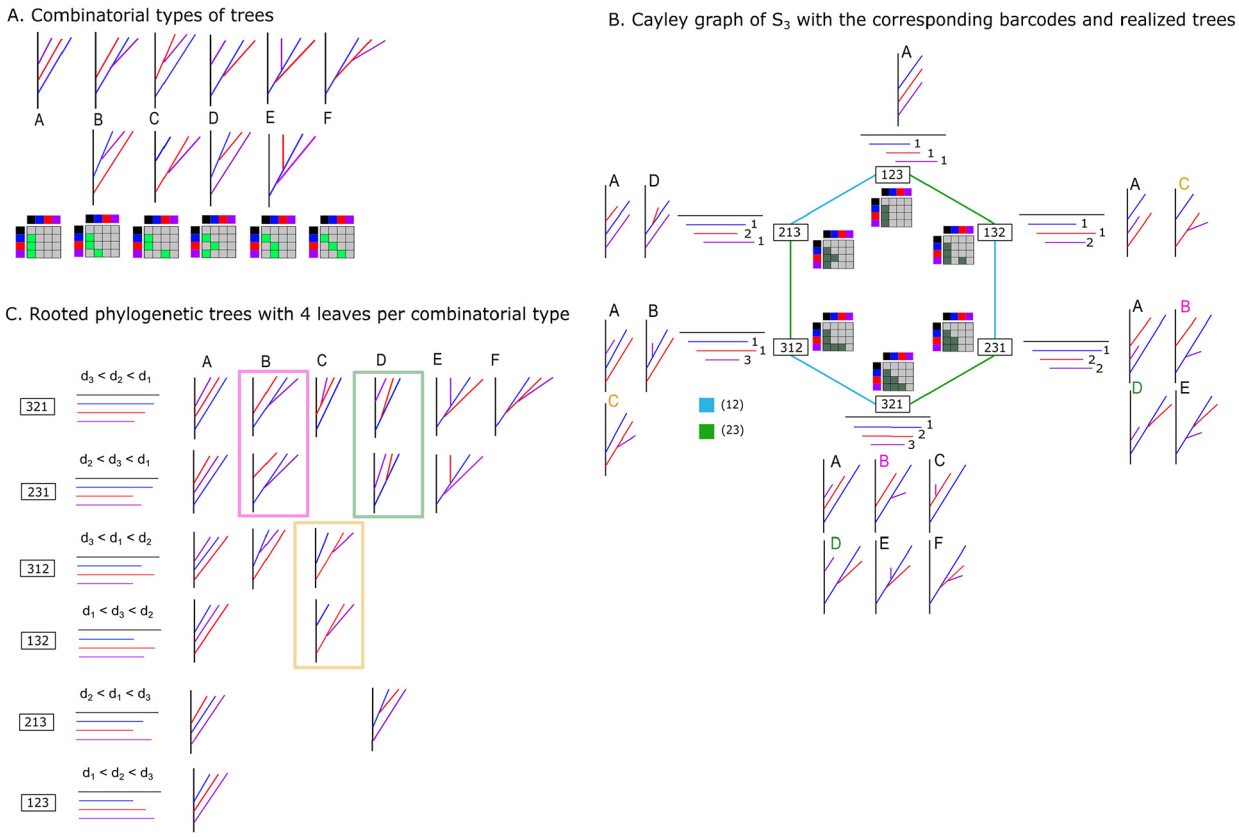


Fig. 6. A. Combinatorial types of rooted trees with four leaves and the corresponding adjacency matrices. B. Cayley graph generated by the two adjacent transpositions of S_3 and the corresponding barcodes, together with all the combinatorial types of trees that realize a barcode. Colored letters correspond to different types of merge trees that are the same as phylogenetic trees (indistinguishable trees), illustrating the result of Section A.1. The number next to each bar corresponds to its index, defined in Proposition 2.18. C. Rooted phylogenetic trees with four leaves. We represent these phylogenetic trees organized by the combinatorial types of barcodes they would have if they had death labels as well. The three pairs of trees within colored squares correspond to the indistinguishable trees defined in Section A.1: the internal nodes are incomparable, so they can have two different death value that lead to different merge trees. In phylogenetic trees, the label order does matter: for instance, in the first column, all the trees are of the same combinatorial type A but correspond to different phylogenetic trees. To go from the space of phylogenetic trees to the space of combinatorial trees, one forgets the labels and considers the adjacencies only, see Fig. 16.

Corollary 2.26. *Let (T,h) be a merge tree and let (T,L_l,L_i) be its corresponding combinatorial merge tree (Remark 2.8). Let B_T be the barcode of (T,h) obtained by applying the Elder Rule on (T,h), and σ_B its associated permutation. If one applies the combinatorial Elder Rule (Definition 2.15), then one the pairing of death labels and birth labels by the Elder rule defines σ_B from (T,L_l,L_i). In other words, the diagram of Fig. 9 commutes.*

Proof. The combinatorial Elder rule applied to (T,L_l,L_i) returns a set of pairs of birth and death labels. To obtain a permutation from it, one considers the pairing induced by the order defined on the death labels (L_i) and the order defined on the birth labels (L_l). By the previous proposition and the definition of the combinatorial Elder rule and the Elder rule, it follows directly that this permutation is the same as σ_B. $\square$

Fig. 9 illustrates the relationship between merge trees and their combinatorial equivalence classes and barcodes and their combinatorial equivalence classes, corresponding to permutations.

3. Combinatorial and algebraic perspectives on the realization number

Now that we have reviewed the basic notions of trees, merge trees, their barcodes, and prior results on the inverse problem detailed in [5] and [18], we are in a position to extend those results. The first observation of this section is that the tree realization number (TRN) of a barcode is simply the product of the entries of the left inversion vector for the permutation associated to a barcode. This is somewhat surprising, as the left inversion vector is a classical object of study, but typically authors study the sum of its entries rather than the product. This observation also allows us to characterize those barcodes that have a larger tree realization number in the language of geometric group theory: permutations that have longer word length in the left Bruhat order have higher TRN. Based on a convexity result for combinatorial equivalence

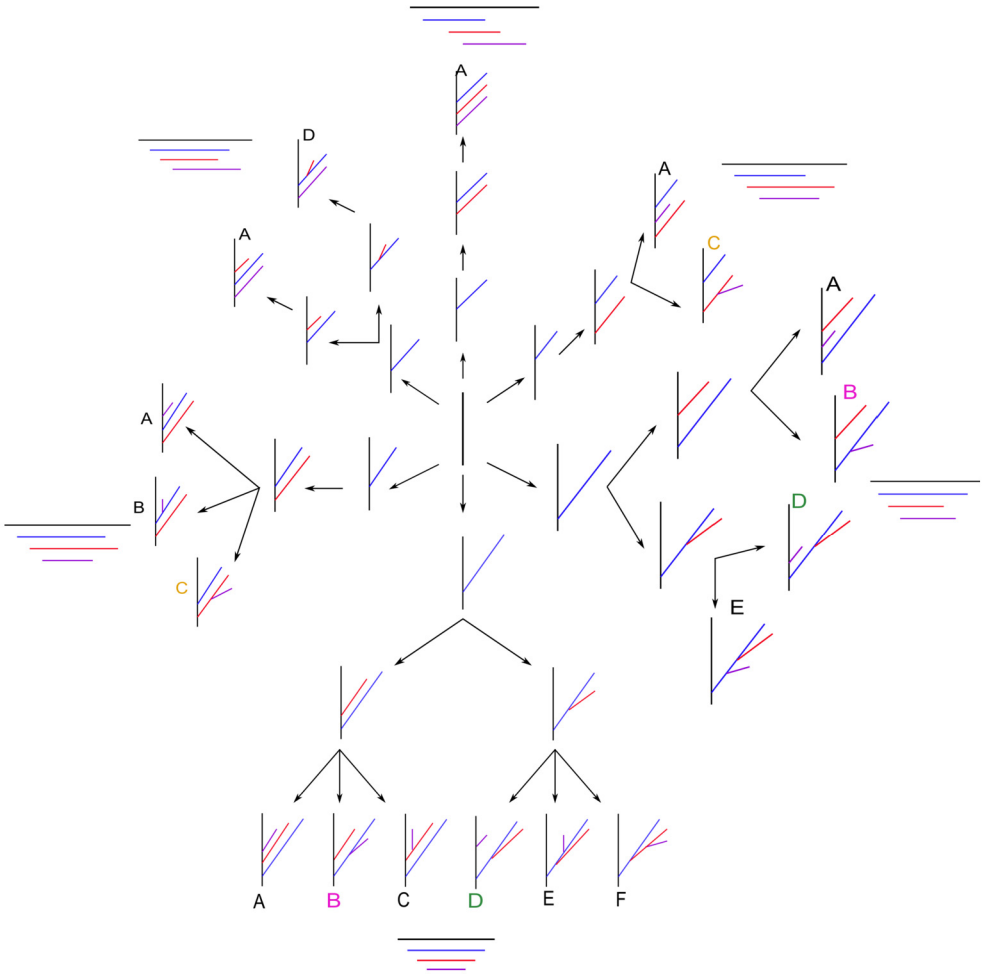


Fig. 7. Recursive construction of all trees realizing barcodes with three non-essential bars. At each bifurcation, the number of new branches corresponds to the index of the branch that is added. Each time we add a new branch, we multiply the number of possibilities by its index, illustrating the result of Proposition 2.18.

classes of barcodes, we also provide a closed form expression for the sum of TRNs across all elements of the symmetric group, which is equal to the number of maximal chains in the lattice of partitions. This result is of use in the next section, when we consider probability distributions on the space of barcodes and calculate the expected tree realization number for the uniform distribution on the symmetric group.

3.1. The realization number and the left inversion vector

Careful inspection of the formula for the tree realization number in Proposition 2.18 reveals that the index of a bar $[b_i, d_i)$ in a barcode B is given by the number of bars born before b_i and that die after d_i. Thinking in terms of the permutation associated to a barcode, this index counts the number of "upsets" of birth-mapping-to-death order. More precisely, for a permutation σ of $\{1, \ldots, n\}$ if $i < j$ and $\sigma(i) > \sigma(j)$, then either the pair of places (i, j) or the pair of elements $(\sigma(i), \sigma(j))$ is called an *inversion* of σ—the usual order $i < j$ has been "upset" or inverted here. We now modify the usual notion of an inversion vector so that it is defined for strict barcodes and makes our theorem statements as tidy as possible.

Definition 3.1. Let $B = [b_0, \infty) \cup \{[b_i, d_i)\}_{i=1}^n$ be a strict barcode with $b_i < b_j$ for $i < j$. The *left inversion vector* of B is the n-vector $l(B)$ whose i-th coordinate is

$$l_i(B) := \#\{j < i \mid d_j > d_i\}.$$

We note that for this formula the index $j = 0$ is used for computation although it is not given a position in the n-vector $l(B)$, since the vector would have length $n + 1$. When we calculate the left inversion vector of a permutation σ associated to a barcode, we use the slightly modified definition

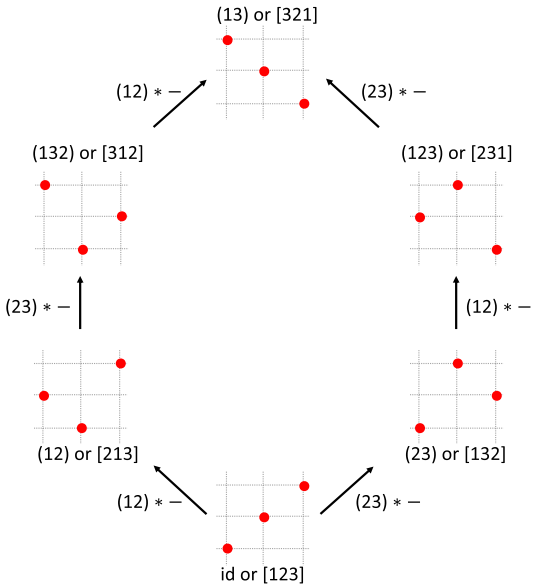


Fig. 8. Combinatorial equivalence classes of persistence diagrams with three non-essential points. The associated permutation σ is written next to each diagram in both forms of notation: the image notation is in square brackets, i.e., $[\sigma(1)\sigma(2)\sigma(3)]$, and the cycle notation in parentheses. The arrows point in the direction of increasing left Bruhat order and exhibit S_3 as a poset. Notice that the permutation acts by switching death order.

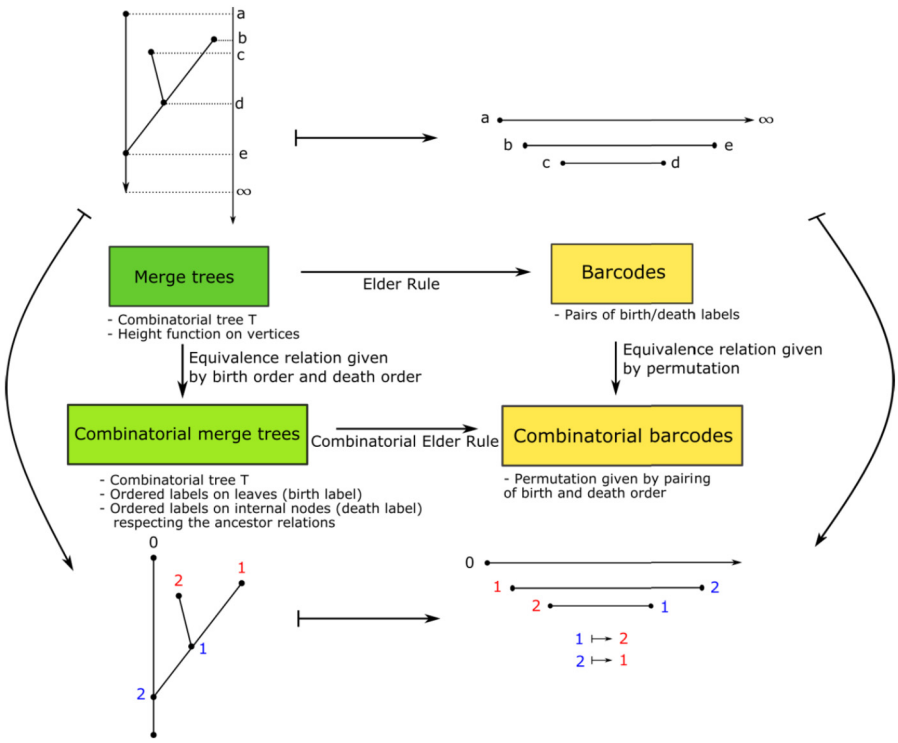
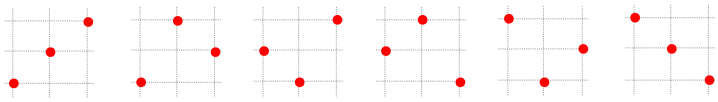


Fig. 9. The relationships between merge trees, combinatorial equivalence classes of merge trees, barcodes and combinatorial barcodes. Birth labels are indicated in red, and death labels in blue. The largest bar (corresponding to the essential class) is not taken into account in the combinatorial setting since it is there for every tree/barcode. Therefore we label it by 0. Considering the pairing of the i-th death and the j-th birth given by the combinatorial Elder rule (bottom right) returns the same permutation as the one directly defined from the barcode (top right).



permutation	id	(23)	(12)	(123)	(132)	(13)
left inversion vector	(1,1,1)	(1,1,2)	(1,2,1)	(1,1,3)	(1,2,2)	(1,2,3)
tree realization number	1	2	2	3	4	6

Fig. 10. Persistence diagrams associated to the six elements of S_3, along with their inversion vector and tree realization number: the i-th position of the inversion vector counts the number of points in the upper left quadrant of the i-th point from the left.

$$l_i(\sigma) := \#\{j \leq i \mid \sigma(j) \geq \sigma(i)\}$$

in order to make sure that $l(\sigma) = l(B)$.

Example 3.2. One can easily compute the left inversion vector of the following barcode with one essential class and four non-essential classes:

$$B = \{[0,\infty), [1,7), [2,6), [3,5), [4,8)\} \quad \Rightarrow \quad l(B) = (1,2,3,1).$$

The permutation associated to this barcode is $\sigma = (3214)$ because the first bar to die corresponds to the third birth, the second to the second, the third to the first and the fourth to the fourth. Clearly, $l(\sigma) = (1,2,3,1)$ as well.

Example 3.3. For the left inversion vectors associated to the six elements of S_3, along with their tree realization numbers, see Fig. 10.

To define coordinates on the space of left inversion vectors, we use the totally ordered sets

$$[k] := \{1 < 2 < \cdots < k\}$$

for k a positive natural number. It is easy to see that the left inversion vector construction establishes a bijective correspondence between S_n and the Cartesian product of sets of the above form, i.e., there is a bijection

$$l : S_n \longrightarrow [1] \times [2] \times \cdots \times [n-1] \times [n] \qquad \text{where} \qquad \sigma \mapsto l(\sigma).$$

The next lemma, which is crucial for the rest of the paper, follows immediately from this observation. It was first established in [18], though not formulated explicitly in terms of the left inversion vector.

Lemma 3.4. *If B is a strict barcode with one essential bar $[b_0,\infty)$ and n non-essential bars $\{[b_i,d_i)\}_{i=1}^n$, then*

$$R(B) = \prod_{i=1}^{n} l_i(\sigma(B)).$$

where $\sigma(B)$ is the permutation associated to the barcode B.

An immediate consequence of this lemma is that if B and B' are combinatorially equivalent barcodes, in the sense of Definition 2.22, then their realization numbers are the same. It follows that the tree realization number induces a function on the symmetric group, i.e.,

$$R : S_n \to \mathbb{N} : \sigma \mapsto \prod_{i=1}^{n} l_i(\sigma).$$

Before analyzing this function on the symmetric group, we identify some interesting properties of the set of barcodes under the combinatorial equivalence relation, to prepare our exploration of the combinatorics of the TRN in earnest in subsequent sections.

3.2. Convexity of combinatorial equivalence classes

In this section we prove that combinatorial equivalence classes are convex in a certain sense: if two strict barcodes B and B' are of the same combinatorial type, then they can be connected by a "line segment" of barcodes[1] all of the same permutation type.

We prove first that the set $\mathcal{B}_n$ admits the algebraic structure necessary to formulate a convexity result.

Lemma 3.5.

(1) *For all $\lambda \in \mathbb{R}_{>0}$ and $B = \{[b_0, \infty)\} \cup \{[b_i, d_i)\}_{i=1}^n \in \mathcal{B}_n$, the set*

$$\lambda B := \{[\lambda b_0, \infty)\} \cup \{[\lambda b_i, \lambda d_i)\}_{i=1}^n$$

is also a strict barcode.

(2) *For all $B = \{[b_0, \infty)\} \cup \{[b_i, d_i)\}_{i=1}^n, B' = \{[b'_0, \infty)\} \cup \{[b'_i, d'_i)\}_{i=1}^n \in \mathcal{B}_n$, the set*

$$B + B' := \{[b_0 + b'_0, \infty)\} \cup \{[b_i + b'_i, d_i + d'_i)\}_{i=1}^n$$

is also a barcode with distinct birth times, which is strict if B and B' have the same permutation type.

Proof. The proof of (1) is trivial, since λ is assumed to be positive, whence multiplication by λ preserves the order of real numbers.

The only subtlety in the proof of (2) concerns distinct death times. If the permutation types of B and B' are different, it could happen that $d_i < d_j$ and $d'_i > d'_j$, but $d_i + d'_i = d_j + d'_j$, so that $B + B'$ would not be strict. If they have the same permutation type, then this cannot happen. □

Lemma 3.6. *For every n and every $\sigma \in S_n$, the set of strict barcodes of permutation type σ is convex, i.e., for B and B' of permutation type σ, the interval*

$$[B, B'] := \{tB + (1-t)B' \mid t \in [0,1]\}$$

is contained in the set of barcodes of permutation type σ.

Proof. Given the previous lemma, it remains only to prove that the permutation type of $tB + (1-t)B'$ is σ, which follows immediately from the observation that

$$d_i < d_j \text{ and } d'_i < d'_j \Longrightarrow d_i + d'_i < d_j + d'_j. \quad \square$$

Remark 3.7. We can also formulate the lemma above as saying that there is a "straight-line path" from B to B',

$$\overline{BB'} : [0,1] \to \mathcal{B}_n : t \mapsto B^t := tB + (1-t)B'.$$

It is not hard to show that this function is indeed continuous with respect to both the bottleneck metric and the Wasserstein metric on $\mathcal{B}_n$, but we choose not to do so here, to avoid introducing further definitions outside of the focus of this paper.

It is interesting also to consider the path $\overline{BB'}$ when the barcodes B and B' are not of the same permutation type. As mentioned in the proof of Lemma 3.5, not every point of $\overline{BB'}$ is necessarily a strict barcode in this case, which allows the path to move from one permutation type to another. One can show that the smallest number of different classes that the path goes through is the length of the shortest path between the two corresponding permutations of B and B' on the Cayley graph defined using the generating set of elementary (neighboring) transpositions $\tau_i = (i, i+1)$. This value is related to the Bruhat order, which we introduce in the next section. The case where two transpositions or more happen simultaneously has been covered in [2], in which the authors give a stratification of the space of barcodes involving Coxeter complexes, polytopes that are dual to the permutohedron.

Example 3.8. Fig. 11 shows an example of the path described in the proof above, using the representation of barcodes as persistence diagrams. The path consists of the straight lines between the matched points of the diagrams. Notice that the order of the indices in $d_1 < d_3 < d_2$ is preserved in $d'_1 < d'_3 < d'_2$ because the barcodes are in the same combinatorial equivalence class. However, comparisons between death or birth values are not preserved. If we moved the red point corresponding to (b_1, d_1) to the right slightly, we could make b_1 greater than b'_2, for example, but this would not effect the equivalence class.

[1] A continuous path of barcodes is sometimes called a *vineyard*. This terminology arises more commonly when barcodes are represented using persistence diagrams, as this path traces out a configuration of points in the plane, with points appearing and disappearing out of the diagonal.

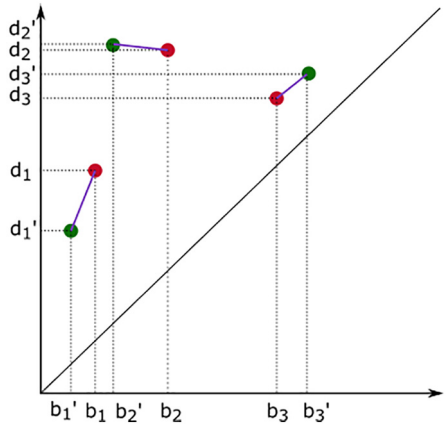


Fig. 11. Continuous path between two barcodes in the same combinatorial class. We show the persistence diagrams of each barcode. The first one B is indicated by red dots and the second one by green dots, and the path B^t is in purple.

Lemma 3.6 allows us to fix a standardized representative of each combinatorial barcode type, making the connection to the symmetric group explicit.

Definition 3.9. A barcode B is in *standard form* if there is a permutation σ of the set $\{1, \dots, n\}$ so that

$$B = \{[i, \sigma(i) + n)\}_{i=1,\dots n} \cup \{[0, \infty)\}.$$

Note that B is strict and has permutation type σ. We will sometimes write $B(\sigma)$ for the standard barcode associated to σ.

Remark 3.10. Barcodes in standard form are in bijection with combinatorial barcode classes, since the latter is in bijection with the symmetric group. Clearly if $\sigma \neq \sigma'$ are different permutations of $\{1, \dots, n\}$, then their corresponding standard form barcodes are different, because $B(\sigma) \neq B(\sigma')$. Moreover, if two barcodes B and B' are in standard form, but $B \neq B'$, then they have different permutation types because there must be an interval $[i, j) \in B$ that is not equal to $[i, j') \in B'$, even though the birth times are the same. This then implies that

$$j \neq j' \Rightarrow \sigma(i) + n \neq \sigma'(i) + n \Rightarrow \sigma(i) \neq \sigma'(i).$$

Finally, Lemma 3.6 implies that any strict barcode B of permutation type σ can be connected via a straight-line path to the barcode $B(\sigma)$ in standard form without changing permutation type.

3.3. Tree realization number preserves Bruhat order

It is interesting to study both the tree realization number from a combinatorial point of view via the symmetric group and the symmetric group from a "barcode" point of view via the realization number. To our knowledge, the product of the components of the left inversion vector is not a very commonly used statistic on symmetric groups, so we take this opportunity to study some of its properties.

Observe first that two adjacent permutations in the Cayley graph (i.e., two permutations that differ by left multiplication by one elementary transposition $\tau_i = (i, i+1)$) never have the same realization number. This follows easily from the definition. As a consequence, the realization number is locally injective, although it is not globally injective, since barcodes of type (12) and type (23) have the same TRN. In this section we extend this local injectivity observation, proving that the TRN defines an order-preserving map from the symmetric group to the natural numbers, when the symmetric group is equipped with the appropriate Bruhat order.

Recall that the symmetric group is generated by elementary transpositions $\tau_i := (i, i+1)$. This implies that any element of S_n can be represented using a word made using the alphabet $\mathcal{A} = \{(i, i+1)\}_{i=1}^{n-1}$, although that representation need not be unique. A word representing a certain permutation is *reduced* if it is of minimal length. The *length* of a permutation is the minimal length of a word representing the permutation.

Definition 3.11 *(Left Bruhat order).* The *left Bruhat order* is a partial order on S_n, specified as follows. If $\sigma, \sigma' \in S_n$, then $\sigma < \sigma'$ if the length of σ is less than that of σ', and there exist $\tau_{i_1}, \dots, \tau_{i_k} \in \mathcal{A}$ such that σ' is represented by a reduced word of the form $\tau_{i_1} \cdots \tau_{i_k} \sigma$.

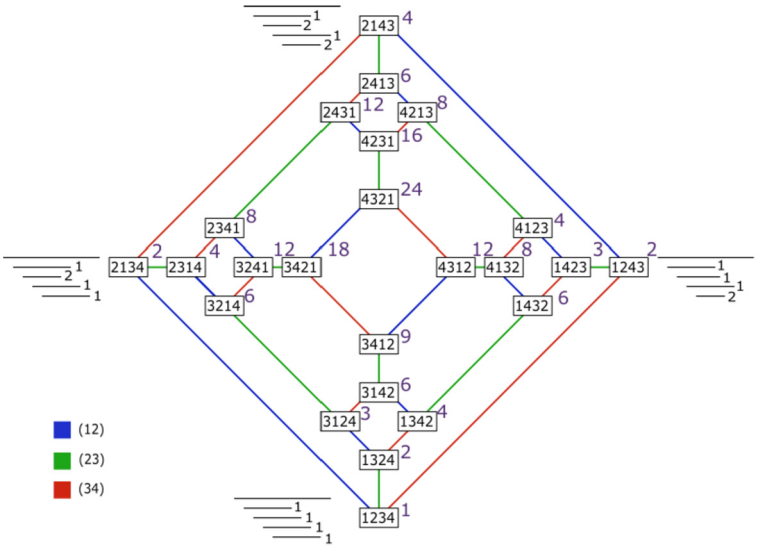


Fig. 12. The Cayley graph of the symmetric group S_4 and some of the corresponding barcodes. We indicate the tree realization number in purple next to each vertex.

Example 3.12. In S_3 we note that $(123) > (23)$ under the left Bruhat order because $(123) = (12)(23)$, where we use cycle notation and where composition is read from right to left. In the left Bruhat order (123) and (12) are not comparable; see Fig. 8.

The next lemma shows that the realization number increases with increasing left Bruhat order. We remark that this lemma can be viewed as a consequence of a classical result, which is mentioned in [9]: if $\sigma < \sigma'$, then the number of inversions in σ' is greater than the number of inversions in σ.

Lemma 3.13. *If $\sigma, \sigma' \in S_n$ are such that $\sigma < \sigma'$ in the left Bruhat order, then $R(\sigma) < R(\sigma')$.*

Proof. Since $\sigma < \sigma'$, there exist $\tau_{i_1}, \ldots, \tau_{i_k} \in \mathcal{A}$ such that $\sigma' = \tau_{i_1} \cdots \tau_{i_k} \sigma$. If $k = 1$, so that σ and σ' are adjacent on the Cayley graph, i.e., $\sigma' = \tau_i \sigma$ for some i. By assumption, the length of σ' is greater than that of σ, which implies that, $\sigma'(i+1) > \sigma'(i)$ while $\sigma(i+1) < \sigma(i)$.

The result now follows from the second assertion in Proposition 3.5 in [18]: translating this result into the language of permutations, we deduce that

$$R(\sigma') = \frac{R(\sigma)(l_{i+1}(\sigma) + 1)}{l_{i+1}(\sigma)} > R(\sigma).$$

The result now follows by induction on the number of transpositions τ_i. $\square$

Example 3.14. One can see the Cayley graph of S_4 in Fig. 12. Notice that two permutations σ, σ' satisfy $\sigma < \sigma'$ in the Bruhat order if and only if a shortest path from σ' to the identity contains a shortest path from σ to the identity. The realization number increases along such paths.

Remark 3.15. It is interesting to consider the TRN as a discrete Morse function [12] on the order complex of S_n. We note that the TRN has a unique max and min on S_n, which conjecturally are the only critical points. If this conjecture was true, it would provide a new proof of a known result, e.g. [9], that the order complex of S_n is homotopy equivalent to a sphere.

3.4. The sum of realization numbers and chains in the lattice of partitions

Given that the tree realization number on the set of strict barcodes induces a function $R : S_n \to \mathbb{N}$, it is natural to study the sum:

$$\sum_{\sigma \in S_n} R(\sigma).$$

As we show in this section, this sum is equal to the number of combinatorial classes of merge trees (Definition 2.7) and provides another quantitative characterization of the difference between merge trees and phylogenetic trees, which is explored further in the next section.

The sum of TRNs also connects this work with a classical object of study in algebraic combinatorics: each combinatorial equivalence class of merge trees corresponds to a maximal chain in the lattice of partitions, ordered by refinement. For topologists this should make intuitive sense: as two connected components merge this coarsens the partition of a sublevel set into connected components. Enumerating these components leads naturally to the study of the partitions of the set of $\{0, 1, \ldots, n\}$.

We start by characterizing the combinatorial equivalence classes of merge trees by introducing several preparatory lemmas.

Lemma 3.16. *If (T, h) and (T', h') are combinatorially equivalent merge trees with associated barcodes B and B', then the straight-line path $\overline{BB'}$ from B to B' lifts to a continuous path (with respect to the interleaving distance) connecting (T, h) and (T', h').*

Proof. Lemma 2.25 guarantees that the barcodes B and B' associated to T and T' have the same permutation type, so that the straight-line path $\overline{BB'}$ of Remark 3.7 does indeed exist, and every point on the path is a barcode of that permutation type by Lemma 3.6. We now apply the Elder Rule to construct a one-parameter family of merge trees

$$[0, 1] \to \mathcal{MT}_n : t \mapsto (T^t, h^t)$$

that lifts the path $\overline{BB'}$.

Since (T, h) and (T', h') are combinatorially equivalent, the trees T and T' are isomorphic as graphs. Without loss of generality, we can suppose that $T = T'$.

To define our one-parameter family of merge trees, we set $T^t = T$ for all $t \in [0, 1]$ and specify the height function $h^t : V(T) \to \mathbb{R} \cup \infty$ as follows. We have no choice but to set $h^t(r) = \infty$, where r is the root, so it remains only to define h^t on the non-root nodes.

If v_i is the i-th leaf node by birth order in T, and therefore corresponds to the i-th bar of B^t, then the $h^t(v_i)$ is chosen to be the birth time of this bar, i.e.,

$$h^t(v_i) = b_i(1 - t) + b_i' t.$$

Similarly, if w_i is the internal node corresponding to the i-th bar in B^t, then $h^t(w_i)$ is chosen to be the death time of this bar, i.e.,

$$h^t(w_i) = d_i(1 - t) + d_i' t.$$

By construction, the barcode associated to (T, h^t) is clearly B^t.

It was shown in [23] (Theorem 2.2) that the interleaving distance between two merge trees is bounded by the maximal difference between the two height functions. Since $T^{t_1} = T^{t_2}$ for all $t_i \in [0, 1]$ and the height functions h^t change continuously with respect to the l_∞ norm, it follows that the path defined by $t \mapsto (T^t, h^t)$ in the space of trees, equipped with the interleaving distance, is continuous. □

Definition 3.17. We say that a generic merge tree (T, h) with $n + 1$ leaves is in *standard form* if its height function h maps its leaf nodes onto $\{0, 1, \ldots, n\}$ and its internal non-root nodes onto $\{n + 1, \ldots, 2n\}$.

Clearly, a merge tree in standard form has a barcode in standard form (Definition 3.9), because the heights of leaf (internal) nodes are exactly the birth (death) times in the barcode. Moreover, combinatorially equivalent merge trees in standard form have the same barcode. To see this, recall that Lemma 2.25 proves that combinatorially equivalent merge trees have combinatorially equivalent barcodes, which, by Remark 3.10, have unique standard form representatives. What is less clear—and at the heart of the next lemma—is whether combinatorially equivalent merge trees with the same barcode are, in fact, isomorphic. This is important because, a priori, the realization number counts isomorphism classes of merge trees and not combinatorial equivalence classes of merge trees.

Lemma 3.18. *If (T, h) and (T', h') are combinatorially equivalent merge trees, then either*

(1) *they are isomorphic as merge trees, written $(T, h) \cong (T', h')$, or*
(2) *they have different barcodes B and B'.*

Note that if the second consequent obtains, then the barcodes will still be combinatorially equivalent, but different in endpoint position.

Proof. We need to show that at least (1) or (2) obtains under the assumptions. We note that the consequent is not vacuously true because there are non-isomorphic merge trees with the same barcode B, which is precisely what the realization number $R(B)$ counts.

Suppose that (2) does not hold, i.e. two combinatorially equivalent merge trees have the same barcode $B = [b_0, \infty) \cup \{[b_i, d_i)\}_{i=1}^n$. We need to show that $(T,h) \cong (T',h')$. By the Elder rule, there are leaf nodes $v_0, v_1, \ldots, v_n \in T$ with $h(v_i) = b_i$ for all $i \in \{0, \ldots, n\}$. Similarly, there are leaf nodes $v'_0, v'_1, \ldots, v'_n \in T'$ with $h'(v'_i) = b_i$. Because (T,h) and (T',h') are combinatorially equivalent, there is a graph isomorphism $\varphi : T \to T'$ that necessarily takes leaf nodes to leaf nodes but without changing their birth order, thus $\varphi(v_i) = v'_i$ for all i. A similar argument holds for internal nodes and their order, thus showing that φ is an isomorphism that fixes the height function, i.e. φ is a merge tree isomorphism (Definition 2.3) and (1) holds.

Now suppose that (1) does not hold under the hypotheses, i.e. there is a combinatorial equivalence φ that changes height of some node in (T,h). This means that the endpoint of some interval in the barcode B' for (T',h') must be different from the barcode B for (T,h), thus (2) holds. We note, although it is not logically necessary that Lemma 2.25 implies that although $B \neq B'$, they are still combinatorially equivalent and have the same permutation type. □

Corollary 3.19. *Combinatorial equivalence classes of strict merge trees are counted by summing the tree realization number $R(\sigma)$ across all $\sigma \in S_n$, i.e.*

$$\sum_{\sigma \in S_n} R(\sigma) = \#\{\textit{combinatorial classes of strict merge trees}\}.$$

Proof. At a high-level we have the following commutative diagram of sets:

$$\begin{tikzcd}
\text{Isomorphism Classes of Strict MTs} \arrow[r, "\chi"] \arrow[d, "\text{Elder Rule}"'] & \text{Combinatorial Classes of Strict MTs} \arrow[d, "\text{Combo Elder Rule}"] \\
\text{Strict Barcodes} \arrow[r, "q"] & \text{Symmetric Group } S_n
\end{tikzcd}$$

The strategy of the proof is to lift the right hand column of the above diagram by embedding it into the left hand column using standard form merge trees and barcodes. We then establish bijections between combinatorial classes of merge trees/barcodes and standard form merge trees/barcodes.

The map χ is well-defined because if two merge trees are isomorphic, then they're combinatorially equivalent, by definition. Moreover, χ is a surjection because every merge tree (perhaps presented with a labeling) belongs to both an isomorphism class and a combinatorial equivalence class. The quotient map q is a surjection by [18].

We define an injective section of q from S_n to the collection of strict barcodes by sending each permutation type σ to its representative strict barcode $B(\sigma)$ in standard form (Definition 3.9). Remark 3.10 already established that q restricts to an injection on standard form barcodes. Moreover, by the Elder rule, every merge tree with barcode $B(\sigma)$ is in standard form (Definition 3.17) and [5,18] showed that the number of (non-isomorphic) merge trees with the barcode $B(\sigma)$ is exactly the realization number $R(\sigma)$.

Lemma 3.16 shows that there is a path from any (labeled) merge tree to one in standard form (Definition 3.17), thus proving that χ restricts to a surjection from the collection of standard form merge trees, written $\mathcal{SFMT}$, to the collection of combinatorial classes of merge trees. It remains to be seen that $\chi|_{\mathcal{SFMT}}$ is also an injection. The contrapositive of Lemma 3.18 says just that: If (T,h) and (T',h') are not isomorphic and have the same barcode, then they can't be combinatorially equivalent either. Since there's one standard form barcode for each permutation type σ, we deduce that χ restricts to a family of injections, one over each $B(\sigma)$, which jointly cover the image of χ. This completes the proof. □

Since every merge tree is combinatorially equivalent to one in standard form, where leaf nodes are at heights $\{0, 1, \ldots, n\}$, we can use this positioning to relate merge trees with maximal chains in the lattice of partitions of n. We review briefly the necessary definitions.

Definition 3.20. A *partition* of the set $\mathbf{n} := \{0, 1, \ldots, n\}$ is a collection of pairwise disjoint subsets $\mathcal{U} = \{U_1, \ldots, U_k\}$ of $\mathbf{n}$ whose union is $\mathbf{n}$. A partition $\mathcal{U}$ *refines* a partition $\mathcal{U}'$, written $\mathcal{U} \preceq \mathcal{U}'$, if every part of $\mathcal{U}'$ is equal to a union of parts of $\mathcal{U}$. Said differently, $\mathcal{U} \preceq \mathcal{U}'$ if for each $U_i \in \mathcal{U}$ there exists $U'_j \in \mathcal{U}'$ such that $U_i \subseteq U'_j$. We denote the set of partitions of $\mathbf{n}$ by $\mathcal{P}_n$. The refinement relation endows the set $\mathcal{P}_n$ with a partial order, which also happens to be a lattice. A *chain* in the lattice of partitions is a sequence of comparable partitions

$$\mathcal{U}_1 \preceq \cdots \preceq \mathcal{U}_\ell.$$

Such a chain is *maximal* if it is not a subsequence of any longer chain.

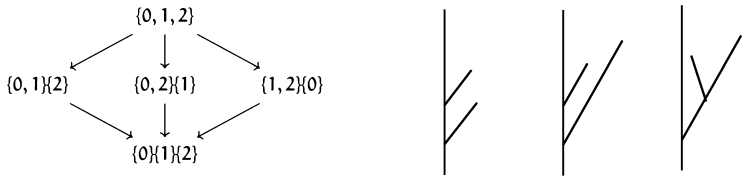


Fig. 13. Left: The lattice of partitions of the set $\{0,1,2\}$. Right: The three possible merge trees corresponding to the maximal chains in the lattice, illustrating Theorem 3.21.

For the sake of notation, we can always write a partition of $\mathbf{n}$ as an ordered list where each subset is separated by a vertical line. The finest possible partition—and hence the bottom element of the $\mathcal{P}_n$—is denoted

$$\{0|1|2|\cdots|n\}.$$

The top element of $\mathcal{P}_n$ is the set $\{\mathbf{n}\}$.

Theorem 3.21. *Combinatorial equivalence classes of merge trees with $n+1$ leaf nodes are in bijective correspondence with maximal chains in the lattice of partitions $\mathcal{P}_n$. As a consequence, the sum of realization numbers is given by the following closed form formula:*

$$\sum_{\sigma\in S_n} R(\sigma) = \sum_{\sigma\in S_n}\prod_{i=1}^{n} l_i(\sigma) = \frac{(n+1)!n!}{2^n}.$$

Proof. Given a merge tree (T,h) in standard form with $n+1$ leaves, we explain first how to construct an associated maximal chain in the lattice of partitions, $\mathcal{P}_n$. We then show that every maximal chain is associated to some merge tree and that non-equivalent trees gives rise to distinct maximal chains.

Since (T,h) is in standard form, all of the merge events (bifurcations) happen after (are at greater height than) all the birth events. It follows that the sublevel set of $h: V(T)\to\mathbb{R}\cup\infty$ at any value in the interval $(n,n+1)\subset\mathbb{R}$ consists of $n+1$ components, corresponding to the finest partition $\mathcal{S}(T)_1 := \{0|1|2|\cdots|n\}$.

As we cross height $n+1$, the definition of the standard form implies that a merge event of two components, born at heights i and j, occurs. This merge event has the effect of coarsening the partition $\mathcal{S}(T)_1$, placing the two elements i and j into a single set of the partition. This defines the next, coarser partition $\mathcal{S}(T)_2$.

In general the i-th partition associated to the tree T is the partition of the leaf nodes into connected components at height $n+i$. At height $2n$ the sublevel set of the tree is connected, which corresponds to the top element in $\mathcal{P}_n$.

Each standard form merge tree thus gives rise to a chain of $2n$ elements in $\mathcal{P}_n$, which is obviously maximal. Moreover, from any maximal chain

$$\mathcal{U}_1 \preceq \cdots \preceq \mathcal{U}_\ell$$

in $\mathcal{P}_n$, one can always build a merge tree that realizes the chain as follows. Start by defining a filtration of the set of subsets of $[n]$, where a subset $V\subset\mathbf{n}$ enters the filtration at $n+i$, where i is the smallest index such that $V\subset U$ for some $U\in\mathcal{U}_i$. This defines a function from the set of subsets of $[n]$ (of which the geometric realization is the n-simplex) to $\mathbb{R}$. Taking the merge tree of this function as in Remark 2.5 associates a merge tree to a chain in $\mathcal{P}_n$.

Injectivity of the map from standard form merge trees to maximal chains is also clear. If two merge trees in standard form produce the same maximal chain, then their heights and adjacency relationships must be the same, i.e., they must be combinatorially equivalent.

The number of maximal chains in $\mathcal{P}_n$ was determined by Erdős and Moon [10] to be $(n+1)!n!2^{-n}$. This number is easily understood in the setting of merge trees. First, one chooses two of the $n+1$ connected components to merge at height $n+1$. Then one chooses two of the remaining n connected components to merge at height $n+2$. This process repeats until we run out of options at height $2n$. The number of ways of constructing standard form merge trees is thus

$$\binom{n+1}{2}\binom{n}{2}\cdots\binom{2}{2} = \frac{(n+1)n}{2}\cdot\frac{n(n-1)}{2}\cdots\frac{2\cdot 1}{2} = \frac{(n+1)!n!}{2^n}. \quad \square$$

Example 3.22. Fig. 13 shows the lattice of partitions on the set $\{0,1,2\}$ together with the three possible merge trees corresponding to the maximal chains in the lattice.

Remark 3.23 *(Expected tree realization number).* It is very convenient that $n!$ appears in the numerator of the sum of realization numbers. As we explain in greater depth in the section on statistics for the realization number, this allows us to compute the average realization number when S_n is equipped with the uniform measure, for which the probability of a

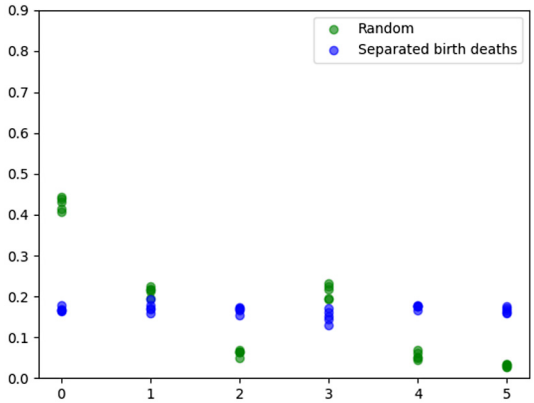


Fig. 14. Two distributions on S_3 induced by distributions of barcodes with three non-essential bars. The elements of S_3 are indexed by the integers $0, \ldots, 5$. Green: we first pick uniformly the birth times b_i in the interval $[0, 100]$, then choose uniformly the death times $d_i \in [b_i, 100]$. Blue: We pick uniformly three birth times $b_i \in [0, 49]$ and three death times $d_i \in [50, 100]$, which induces a uniform distribution on the symmetric group S_3.

permutation σ $P(\sigma) = \frac{1}{n!}$. Indeed, by rearranging terms slightly, we see that the expected realization number is determined by the ratio of $(n+1)!$ and 2^n:

$$\mathbb{E}[R] = \sum_{\sigma \in S_n} R(\sigma) P(\sigma) = \frac{1}{n!} \frac{(n+1)!n!}{2^n} = \frac{(n+1)!}{2^n}.$$

Before studying the probabilistic aspects of the realization number more fully, we first compare Theorem 3.21 with analogous counting results for phylogenetic trees in the next section.

4. The probabilistic study of tree realization numbers

As already foreshadowed by Remark 3.23, the formula in Theorem 3.21 provides us with an unexpected gift in the study of statistics for realization numbers. Assuming that every combinatorial type of barcode is equally likely, so that each permutation type σ has probability $\frac{1}{n!}$, we calculated that the expected tree realization number (TRN) is

$$E[R] = \sum_{\sigma \in S_n} R(\sigma) P(\sigma) = \frac{1}{n!} \frac{(n+1)!n!}{2^n} = \frac{(n+1)!}{2^n}.$$

We regard the assumption that each barcode permutation type is equally likely as a sort of "null hypothesis" to be tested against. Even if one considers Gaussian perturbations to functional data, characterizing the image of this measure on the space of merge trees and hence (combinatorial types) of barcodes is an open problem. Depending on the setup, it may be the case that features tend to die in the order in which they are born (a sort of "topological first in first out" queue) or it might be the case that features die in the opposite order in which they are born (a "first in last out" queue). In general, for real data, it is unlikely that the distribution of permutation types of (barcodes of) merge trees will be uniform. Regardless, characterizing the distribution of TRNs in terms of the output of the function $R : S_n \to \mathbb{N}$ when S_n is equipped with the uniform measure provides an important null hypothese against which to test real data.

In this section we start with a brief outline of computational methods for generating random barcodes and compare the corresponding distribution of permutation types with the uniform distribution. We then provide formulas for first and second moments of the pushforward distribution $\pi_n := R_* \mu_n$, where μ_n is the uniform measure on S_n. This allows us to calculate the variance of the TRN, which opens the door to hypothesis testing wherever the map from trees to barcodes is of interest to scientific applications.

Somewhat surprisingly, Theorem 4.1 says that the exact value for the measure π_n can be determined from π_{n-1} and Dirichlet convolution with the uniform distribution on S_{n-1}, enabling us to study the entire distribution of TRNs as the number of features varies. To conclude, we provide a novel closed-form formula for the expected log-realization number, which allows us to characterize the empirical data in Fig. 2 in a more analytical manner.

4.1. Distributions of randomly generated barcodes

In this section we briefly describe two methods to generate random barcodes and consider the pushforward distribution on S_n for each of these. This pushforward is defined by the identification of barcodes with permutations as described above.

The first method was used in [18] to generate barcodes and compare their realization numbers to the ones of biological barcodes, in a way similar to Fig. 2. To generate a barcode with n bars, for each bar we first pick a birth time b_i uniformly at random in the interval $[0, 100]$ and then pick a death time $d_i \in [b_i, 100]$ uniformly at random. Because the latter distribution is conditioned on $d_i > b_i$, the induced distribution on the symmetric group is not uniform, as seen in Fig. 14 with the "random" green dots.

The second method displayed in Fig. 14 forces separation of births and deaths to guarantee a uniform distribution on the symmetric group. To generate n bars in a barcode, we first choose n births uniformly in the interval $[0, 49]$, then n death times d_i uniformly in $[50, 100]$. A moment of reflection shows that this provides a uniform distribution on S_n, as seen in Fig. 14 with the "separated" blue points.

4.2. The distribution of tree realization numbers via Dirichlet convolution

Let μ_n denote the uniform distribution on S_n. By our correspondence, this is also a distribution on combinatorial equivalence classes of barcodes. The tree realization number $R : S_n \to \mathbb{N}$ then defines a random variable where the probability $P(R = t)$ is determined by the number of permutations n_t with realization number t. The following theorem states that this probability can be computed recursively via convolution with the uniform distribution on $1, \ldots, n$.

Theorem 4.1. *For any $k \geq 1$, let μ_n denote the uniform distribution on S_n and $\pi_n = R_*(\mu_n)$ its pushforward onto $\mathbb{N}$ via $R : S_n \to \mathbb{N}$. For any $k \in \mathbb{N}$, let U_k denote the uniform distribution on $\{1, 2, ..., k\}$.*

The probability mass function of π_n can be recursively defined as follows.

- $\pi_1 = U_1$.
- *For $k > 1$, $\pi_k = U_k * \pi_{k-1}$, where $*$ indicates Dirichlet convolution, i.e.,*

$$\pi_k(c) = \sum_{ab=c} U_k(a)\pi_{k-1}(b) \text{ for all } c \in \mathbb{N}.$$

It follows immediately from this theorem that

$$\pi_n = U_n * U_{n-1} * \ldots * U_1$$

for all $n \geq 1$.

Proof. We prove this theorem by induction on k. It holds trivially for $k = 1$. Suppose that it holds for $k - 1$ for some $k \geq 2$. Each number that has positive probability under π_{k-1} corresponds to $R(\sigma) = \prod_{i=1}^{k-1} l_i(\sigma)$ for some $\sigma \in S_{k-1}$.

Consider the map $\kappa_{k-1}^j : S_{k-1} \to S_k$ that embeds S_{k-1} into S_k as follows. For every $\sigma \in S_{k-1}$, the permutation $\kappa_{k-1}^j(\sigma)$ is specified by

$$\kappa_{k-1}^j(\sigma)(i) = \begin{cases} j & \text{if } i = k \\ \sigma(i) + 1 & \text{if } \sigma(i) \geq j \\ \sigma(i) & \text{if } \sigma(i) < j. \end{cases}$$

In other words, κ_{k-1}^j sends $\sigma \in S_{k-1}$ to the permutation $\kappa_{k-1}^j(\sigma) \in S_k$ that maps the k-th object to j and then "bumps up" by one the assigned value of elements in $\{1, 2, ..., k-1\}$ that are mapped to an element greater than or equal to j.

Each map in the collection $\{\kappa_{k-1}^j\}_{j=1}^k$ is injective and collectively their images surject onto S_k. To determine the realization numbers for S_k, we therefore need only compute the realization number of $\kappa_{k-1}^j(\sigma)$ for all $j \in \{1, \ldots, k\}$ and $\sigma \in S_{k-1}$.

Consider $R\big(\kappa_{k-1}^j(\sigma)\big) = \prod_{i=1}^k l_i\big(\kappa_{k-1}^j(\sigma)\big)$. Since $l_i(\sigma) = |\{r \leq i \mid \sigma(r) \geq \sigma(i)\}|$ for any permutation σ, it follows that

$$l_i\big(\kappa_{k-1}^j(\sigma)\big) = l_i(\sigma)$$

for all $i < k$. On the other hand, since $r \leq k$ for all $r \in \{1, .., k\}$,

$$|\{r \leq k \mid \kappa_{k-1}^j(\sigma)(r) \geq \kappa_{k-1}^j(\sigma)(k)\}| = |\{r \mid \sigma(r) \geq j\}| = k - j + 1.$$

We conclude that $R\big(\kappa_{k-1}^j(\sigma)\big) = (k - j + 1)R(\sigma)$.

By the construction of κ_{k-1}^j,

$$\mu_k = \frac{1}{k} \sum_{j=1}^k (\kappa_{k-1}^j)_*(\mu_{k-1}),$$

where $(\kappa_{k-1}^j)_*(\mu_{k-1})$ is the pushforward of μ_{k-1} by κ_{k-1}^j, since each pushforward assigns mass $\frac{1}{(k-1)!}$ to each element of a unique subset of size $(k-1)!$ in S_k.

We are now prepared to compute π_k. Let $x \in \mathbb{N}$.

$$\pi_k(x) = R_*(\mu_k)(x) = \mu_k\big(R^{-1}(x)\big) \tag{1}$$

$$= \frac{1}{k}\sum_{j=1}^{k}(\kappa_{k-1}^j)_*(\mu_{k-1})\big(R^{-1}(x)\big) \tag{2}$$

$$= \frac{1}{k}\sum_{j=1}^{k}\mu_{k-1}\Big((\kappa_{k-1}^j)^{-1}\big(R^{-1}(x)\big)\Big) \tag{3}$$

$$= \frac{1}{k}\sum_{j=1}^{k}\mu_{k-1}\Big((\kappa_{k-1}^j)^{-1}\big(\{\sigma \in S_k \mid R(\sigma) = x\}\big)\Big) \tag{4}$$

$$= \frac{1}{k}\sum_{j=1}^{k}\mu_{k-1}\big(\{\tilde{\sigma} \in S_{k-1} \mid (k-j+1)\cdot R(\tilde{\sigma}) = x\}\big) \tag{5}$$

$$= \frac{1}{k}\sum_{j=1}^{k}\mu_{k-1}\big(\{\tilde{\sigma} \in S_{k-1} \mid j\cdot R(\tilde{\sigma}) = x\}\big) \tag{6}$$

$$= \frac{1}{k}\sum_{jb=x}\mu_{k-1}\big(\{\tilde{\sigma} \in S_{k-1} \mid R(\tilde{\sigma}) = b\}\big)\mathbb{1}_{[k]}(j) \tag{7}$$

$$= \sum_{ab=x}\mu_{k-1}\big(\{\tilde{\sigma} \in S_{k-1} \mid R(\tilde{\sigma}) = b\}\big)\frac{\mathbb{1}_{[k]}(a)}{k} \tag{8}$$

$$= \sum_{ab=x}U_k(a)\pi_{k-1}(b), \tag{9}$$

where the second line follows from the identity $\mu_k = \frac{1}{k}\sum_{j=1}^{k}(\kappa_{k-1}^j)_*(\mu_{k-1})$, the fifth line follows from $R(\kappa_{k-1}^j(\sigma)) = (k-j+1)R(\sigma)$, and the sixth and seventh lines are simple changes of variables. □

For what follows, it is useful to consider for each n the multiset Π_n, which is the range of $R : S_n \to \mathbb{N}$, taking into account multiplicities. Let $m_n : \mathbb{N} \to \mathbb{Z}_{\geq 0}$ be the multiplicity function of Π_n, i.e., $m_n(x)$ is the number of times $x \in \mathbb{N}$ appears in Π_n, which is the number of permutations in S_n that have realization number x. In particular, $m_n(x) = 0$ if and only if $x \notin \Pi_n$.

Since π_n is the pushforward of the uniform distribution on S_n, the probability of each x is determined by dividing the multiplicity function by $n!$, i.e., $\pi_n(x) = \frac{m_n(x)}{n!}$. The following corollary follows directly from the construction of π_n.

Corollary 4.2. *The multiset Π_n can be constructed recursively as follows:*

- $\Pi_1 = \{1\}$.
- *For $i > 1$, Π_i is the multiset with multiplicity function specified by*

$$m_n(x) = \sum_{ab=x} m(b)\mathbb{1}_{[i]}(a)\mathbb{1}_{\Pi_{i-1}}(b).$$

In other words, Π_n can be defined as a $[k] * \Pi_{n-1}$, where $[k] = \{1, \ldots, k\}$ and $*$ is the Dirichlet convolution of multisets.

Example 4.3. We now explicitly describe Π_i for $i \in \{1,2,3,4\}$. For convenience we write the multisets Π_i as sets with repetition. Counting the number of appearances of a number k determines $m_i(k)$.

$$\Pi_1 = \{1\} \tag{10}$$

$$\Pi_2 = \{1,2\} \tag{11}$$

$$\Pi_3 = \{1,2,2,4,3,6\} \tag{12}$$

$$\Pi_4 = \{1,2,2,2,3,3,4,4,4,4,6,6,6,6,8,8,8,9,12,12,12,16,18,24\} \tag{13}$$

The distribution of Π_n is shown in Fig. 15 for $n = 7,8,9$.

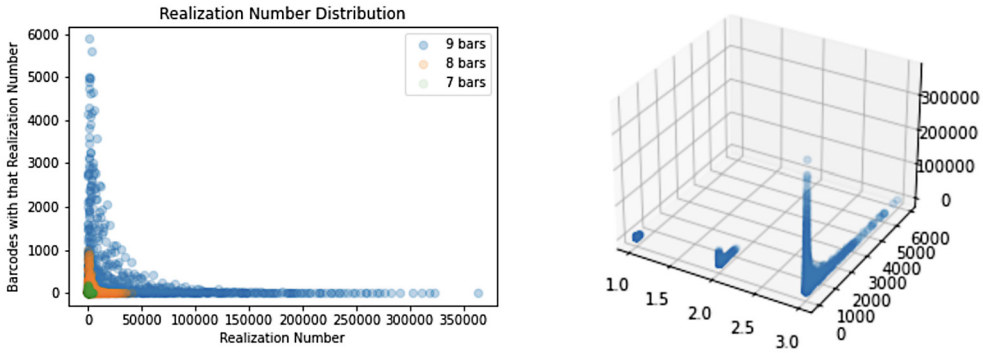


Fig. 15. Distribution of realization numbers for 7,8,9 bars.

To conclude this section, we consider the moments of π_n. We explicitly calculate its first and second moments, obtaining the mean and variance of π_n as corollaries, and outline a general formula for the higher moments. Recall that by knowing these two moments one can use Chebyshev's inequality to calculate the probability of observing some deviation from the mean, an important first pass at hypothesis testing.

Proposition 4.4. $\mathbb{E}(\pi_n) = \frac{(n+1)!}{2^n}$.

Proof. This is the content of Remark 3.23, which establishes this proposition as a consequence of Theorem 3.21. □

Proposition 4.5. $\mathbb{E}(\pi_n^2) = \frac{(2n+1)!}{12^n}$.

Proof. We prove the result by induction on n. The base case ($n = 1$) holds trivially, so assume that the formula holds for $l \leq k$, for $k \geq 1$.

Consider $\mathbb{E}(\pi_{k+1}^2) = \frac{1}{(k+1)!}\sum_{b\in B_{k+1}} R(b)^2$. From the fact that $\pi_{k+1}(x) = \frac{m_{k+1}(x)}{(k+1)!}$, we get:

$$\sum_{b\in B_{k+1}} R(b)^2 = (k+1)!\mathbb{E}(\pi_{k+1}^2) = \sum_{x\in\mathbb{N}} m_{k+1}(x)^2,$$

to prove our result, we need only show that

$$\sum_{x\in\mathbb{N}} m_{k+1}(x)^2 = \frac{((k+1)+1)!(2(k+1)+1)!}{12^{k+1}}.$$

We call the quantity on the left $\mathbb{E}(\Pi_{k+1}^2)$:

$$\mathbb{E}(\Pi_{k+1}^2) = \sum_{b\in B_{k+1}} R(b)^2 = \sum_{a=1}^{k+1}\sum_{b\in B_k}(aR(b))^2 = \sum_{a=1}^{k+1} a^2 \sum_{b\in B_k} R(b)^2 = \sum_{a=1}^{k+1} a^2\mathbb{E}(\Pi_k^2).$$

By the sum of squares formula, we can rewrite this as

$$\begin{aligned}
&\left(\frac{(k+1)(k+2)(2k+3)}{6}\right)\mathbb{E}(\Pi_k^2)\\
&= \left(\frac{(k+1)(k+2)(2k+3)(2k+2)}{(2k+2)6}\right)\mathbb{E}(\Pi_k^2)\\
&= \left(\frac{(k+2)(2k+2)(2k+3)}{2*6}\right)\mathbb{E}(\Pi_k^2)\\
&= \left(\frac{(k+2)(2k+2)(2k+3)}{12}\right)\left(\frac{(k+1)!(2k+1)!}{12^k}\right)\\
&\frac{(k+2)!(2k+3)!}{12^{k+1}} = \frac{((k+1)+1)!(2(k+1)+1)!}{12^{k+1}}. \quad \square
\end{aligned}$$

Corollary 4.6. $\mathbb{V}(\pi_n) = \mathbb{E}(\pi_n^2) - \mathbb{E}(\pi_n)^2 = \frac{1}{n!}\left(\frac{(n+1)!(2n+1)!}{12^n} - \frac{(n!(n+1)!)^2}{n!4^n}\right)$.

Remark 4.7 *(Higher moments of the TRN).* In general, we can define the k-th moment $\mathbb{E}(\pi_n^k)$ by rewriting $n!\mathbb{E}(\pi_n^k) = \mathbb{E}(\Pi_n^k) = (\sum_{a=1}^n a^k)\mathbb{E}(\Pi_n^{k-1})$ and using this recursive relationship to compute a formula. We note that by Faulhaber's formula,

$$\sum_{a=1}^{n} a^k = \sum_{i=0}^{k} \frac{(-1)^{k-i}}{i+1}\binom{k}{i} B_{k-i} n^{i+1},$$

where B_{k-i} is the $k-i$ Bernoulli number.

One can view the results above as a complete characterization of TRNs under the null hypothesis that combinatorial classes of barcodes are distributed uniformly or as part of the growing literature on statistics on the symmetric group, see e.g., [19]. In the following section, we investigate another such statistic.

4.3. Distributions of log realization numbers

Since the maximum realization number for a barcode with n non-essential bars is $n!$, it is convenient to work instead with the logarithm of the realization number, which we call the log realization number. The log realization number was used in [18] as a statistic on barcodes obtained from dendrites; see Fig. 2 for a reminder. This was shown to distinguish between apical and cortical dendrites. Of course, the process of taking the logarithm affects the distribution of TRNs. Jensen's inequality provides a way to bound the expected log realization number. In this section we compute the expected log realization number of uniformly drawn barcodes.

Proposition 4.8. *The expected log realization number for a combinatorial class B of barcodes drawn from the uniform distribution on S_n is*

$$\mathbb{E}_{\mu_n}\Big(\log\big(R(B)\big)\Big) = \sum_{i=1}^{n} \frac{\log(i!)}{i}.$$

Proof. Recall that the set of left inversion vectors can be coordinatized as $[1]\times[2]\times\ldots\times[n]$. Since this Cartesian product has size $n!$, a uniform distribution on S_n can be viewed as a uniform distribution on the set of left inversion vectors. The notation $\mathbb{P}(B\sim\mu_n)$ denotes the probability of a combinatorial equivalence class of barcodes B under the uniform distribution, that is $\frac{1}{n!}$. It follows that

$$\begin{aligned}
\mathbb{E}_{\mu_n}\Big(\log\big(R(B)\big)\Big) &= \sum_{B\in S_n} \log\Big(\prod_{i=1}^{n} l_i(B)\Big)\mathbb{P}(B\sim\mu_n) \\
&= \frac{1}{n!}\sum_{B\in[1]\times\ldots\times[n]}\ \sum_{i=1}^{n} \log(l_i(B)) \\
&= \frac{1}{n!}\sum_{i=1}^{n}\ \sum_{B\in[1]\times\ldots\times[n]} \log(l_i(B)).
\end{aligned}$$

Since $B\sim\mu_n$, and each coordinate in $[1]\times[2]\times\ldots\times[n]$ is independent, the interior sum (for fixed i) is equal to $\frac{n!}{i}(\log(1)+\log(2)+\ldots+\log(i))$. Hence

$$\mathbb{E}_{\mu_n}(\log(R(b))) = \sum_{i=1}^{n} \frac{\log(1)+\log(2)+\ldots+\log(i)}{i} = \sum_{i=1}^{n} \frac{\log(i!)}{i}. \quad \square$$

5. Conclusion

In this paper, we extended the results of [18] and [5] to provide a more precise characterization of the distribution of tree realization numbers (TRNs). This investigation led us to consider the uniform distribution on the symmetric group and the expected TRN, which in turn put us in a setting where classical results from combinatorics could be used. This extraction of the notion of a combinatorial version of a merge tree led us to understand more precisely the difference between merge trees and metric phylogenetic trees [1].

We emphasize that the TRN provides a convenient summary statistic on the space of barcodes that could lead to a better understanding of inherent biological properties of neurons. If we can identify where biological barcodes live on the space of barcodes, it opens the door to many applications such as statistics of learning of "biological" barcodes, allowing to create artificial barcodes that mimic the properties of biological ones and hence to generate neurons from them that are

statistically relevant, yet express higher variability. By studying the simplest possible version of a null hypothesis—where combinatorial equivalence classes of barcodes are uniformly distributed—we are in a position to move on to study more interesting variants on the null hypothesis in TDA and explore the geometry of barcode space in even greater detail.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgements

JC would like to acknowledge NSF Grant CCF-1850052 and NASA Contract 80GRC020C0016 for supporting his research. LK was supported by funding to the Blue Brain Project, a research center of the École polytechnique fédérale de Lausanne (EPFL), from the Swiss government's ETH Board of the Swiss Federal Institutes of Technology. AG and KH gratefully acknowledge the support of Swiss National Science Foundation, Grant No. CRSII5_177237.

Appendix A. Different spaces of trees

As mentioned earlier, most of the language concerning trees is inspired by the study of ancestral relationships. Although trees have been used for this purpose for centuries, a formal definition of a phylogenetic tree—and more importantly a clear coordinatization on the set of all phylogenetic trees—was given only somewhat recently in the landmark paper of Billera, Holmes and Vogtmann [1]. We review some of these definitions, modifying the terminology slightly for our purposes.

Definition A.1. A *metric phylogenetic tree* is a rooted combinatorial tree T endowed with

(1) a labeling of the leaf nodes, and
(2) a non-negative real number associated to every parent-child pair.

The values assigned to each parent-child pair can be considered as weights on the graph edges. By contrast, a *combinatorial phylogenetic tree* is a rooted combinatorial tree with just a labeling of the leaf nodes. If we say *phylogenetic tree* without any modifier, we always mean a combinatorial phylogenetic tree.

Example A.2. Fig. 6B shows all combinatorial classes of merge trees with four leaves and Fig. 6C shows all combinatorial classes of phylogenetic trees with four leaves.

One of the key differences between metric phylogenetic trees and merge trees is that phylogenetic trees always have labeled leaf nodes, with labels independent of the lengths on the edges. This makes sense because *BHV space*—the set of all possible metric phylogenetic trees on n leaf nodes, denoted $\mathcal{MPT}_n$—documents all possible evolutionary relationships among n fixed species. The labels matter because the involved species matter.

On the other hand, the set of all merge trees with n leaf nodes, written $\mathcal{MT}_n$, consists of isomorphism classes of merge trees, see Definition 2.3. We consider also the set $\mathcal{LMT}_n$ of labeled merge trees with n leaves, where the labeling is arbitrary (see Definition 2.1). Let

$$\mathcal{I} : \mathcal{LMT}_n \longrightarrow \mathcal{MT}_n,$$

denote the map that sends a labeled merge tree to its isomorphism class.

We describe the relationship between these two types of tree spaces in the following proposition.

Proposition A.3. *For every $\Delta \in \mathbb{R}$, there is an injective map from the set of metric phylogenetic trees with n leaves, $\mathcal{MPT}_n$, to the set of labeled merge trees with n leaves, $\mathcal{LMT}_n$*

$$\mathcal{H}_\Delta : \mathcal{MPT}_n \longrightarrow \mathcal{LMT}_n.$$

such that the composite $\mathcal{I} \circ \mathcal{H}_\Delta$ has a fiber of cardinality $n!$ over generic merge trees, corresponding to permutations of the labels on the leaf nodes. Moreover, if $\Delta \geq 0$, there is a natural map

$$\mathcal{T}_\Delta : \mathcal{MT}_{n,generic} \longrightarrow \mathcal{MPT}_n$$

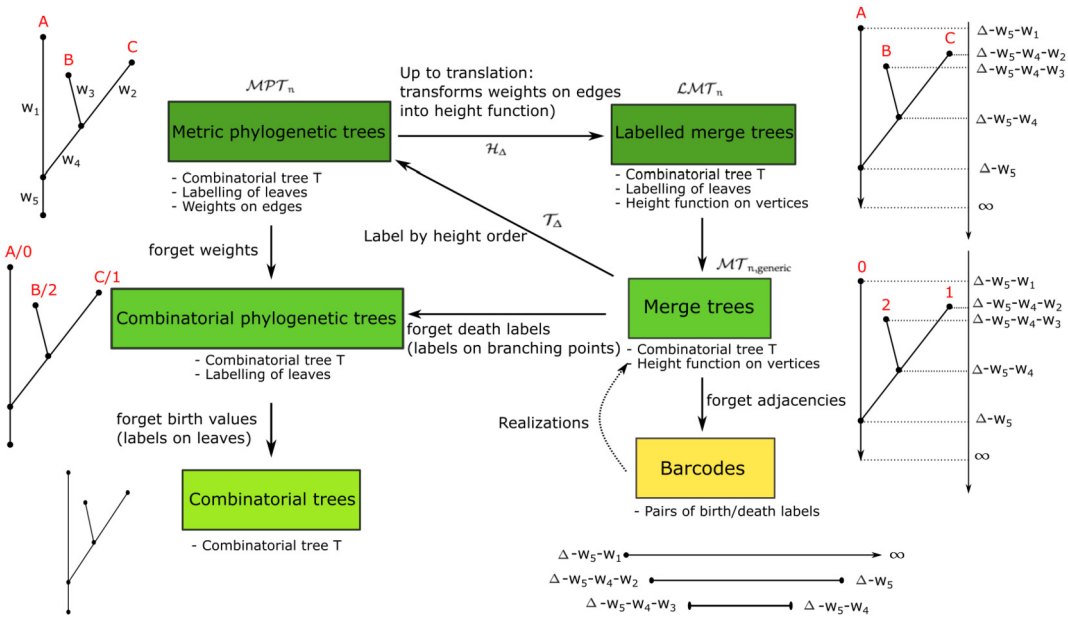


Fig. 16. Summary of the different notions studied in this paper and their relations, as expressed in part by Proposition A.3. One can turn a metric phylogenetic tree (with labels A,B,C in red) into a labeled merge tree. Generic merge trees can be turned into metric phylogenetic trees by labeling according to birth order (labels $0, 1, 2$ in red), but this introduces a discontinuity.

that sends a generic merge tree to a metric phylogenetic tree that is labeled by birth order and where the distance from the root node to its child is Δ.

Proof. Given a metric phylogenetic structure on a rooted tree T, we can define a height function h on T as follows. Every node v that is not the root node r is assigned the function value $h(v) := \Delta - d(r, v)$, where d is the sum of the weights of each edge along the unique path connecting r to v. This defines the map $\mathcal{H}_\Delta$ in the statement of the proposition.

As explained earlier, every generic merge tree admits a canonical ordering of its leaf nodes by height order. If two generic labeled merge trees in the image of $\mathcal{H}_\Delta$ are isomorphic as merge trees, then there is a unique permutation of the n leaf labels taking one labeling to the other. This proves the second statement.

Finally, the map $\mathcal{T}_\Delta$ sends a generic unlabeled merge tree (T, h) to the metric phylogenetic structure on T that has labels given by birth order and where the weight on an edge is given by the difference in heights of its two vertices. The distance from the root node to its child is given by Δ. $\square$

Remark A.4. Each of the three sets above can be equipped with topologies. In [1], the space of phylogenetic trees is topologized as a CAT(0) space where each orthant records a distinct *split topology*. Both labeled merge trees and merge trees can be topologized using versions of the interleaving distance [28]. Unfortunately, the map $\mathcal{T}_\Delta$ is discontinuous with respect to these topologies, as can be seen from Fig. 5.

Proposition A.3 shows that, despite their apparent similarity, there are significant differences between metric phylogenetic trees and merge trees. Indeed neither of the maps above is a bijection. However, if one quotients the set of labeled merge trees by translations, then the map induced by $\mathcal{H}_\Delta$ should be a bijection; alternatively one could modify the definition of merge trees so that the root node has a fixed height N, as in the drawing convention of Remark 2.4.

Although the proposition and remark above identify certain differences and similarities between metric phylogenetic trees and merge trees, for this paper the most important distinction is in terms of combinatorial type. In this respect, merge trees and phylogenetic trees are distinguished by the explicit ordering of birth and death nodes. This observation leads to different formulas for the number equivalence classes (top-dimensional strata in the space) of phylogenetic trees $\mathcal{PT}_n$, which is $(2n-3)!!$, and in $\mathcal{MT}_n$, which is $(n-1)!n!2^{-n+1}$. For now, however, the reader is encouraged to consult Table 1 and Fig. 16 for two convenient summaries of the similarities and differences between combinatorial trees, merge trees, (combinatorial) phylogenetic trees, and barcodes.

A.1. Counting merge trees versus phylogenetic trees

In this section, we compare two counting results for combinatorial merge trees and for phylogenetic trees. On the one hand, Theorem 3.21 implies that there are $\dfrac{(n+1)!n!}{2^n}$ different combinatorial merge trees with $n+1$ leaves. On the other

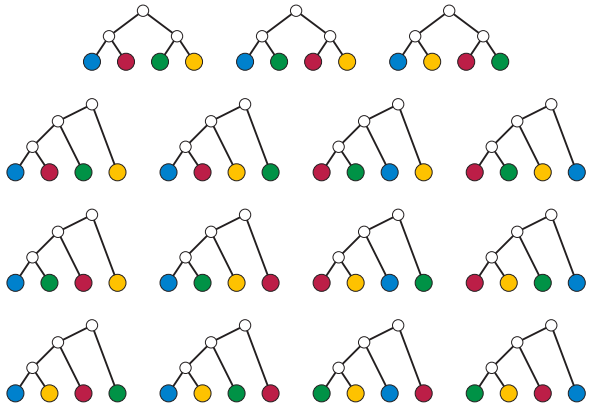


Fig. 17. [Wikipedia, "Double Factorial", "Unordered binary trees with 4 leaves", n.d.]. The 15 different binary rooted trees with four labeled-by-color leaf nodes. The top node should be regarded as the unique child of the root node. This should be compared with the 18 different merge trees in Fig. 6C, as discussed in Example A.5.

hand, it was shown in [11] that there are $(2n-1)!!$ distinct combinatorial phylogenetic trees with $n+1$ leaves. In general, there are more classes of merge trees than there are phylogenetic trees. In the next example, we work through the case $n=3$ in detail.

Example A.5. For $n=3$, i.e., 4 leaf nodes, these formulas imply that there are 18 different classes of merge trees, but only 15 classes of phylogenetic trees, shown in Fig. 17. In Fig. 6C, one can see the 18 different classes of merge trees, arranged by row according to their permutation type in S_3. There are three pairs of merge trees highlighted with colored boxes that correspond to the same combinatorial type of phylogenetic tree.

As the example above shows, the essential difference between classes of merge trees and classes of phylogenetic trees is that merge trees are sensitive to relative heights of internal (death) nodes, whereas a phylogenetic tree is not. This also explains why two combinatorially equivalent metric phylogenetic trees (T,m) and (T',m') may be associated to different permutation types, if one uses Proposition A.3 to define a height function on each and compute a barcode according to the Elder rule. However there are certain orders of births and deaths that must be preserved. As one can see in Fig. 6C, the pair of trees in the purple box under column B both have the blue bar being born before and dying after the purple bar; the relative positioning of the death time associated to the red bar is the only thing that changes.

In this section we pinpoint more precisely how many different classes of merge trees can produce the same class of phylogenetic tree. As one might imagine, this is dictated in part by certain subgroups of the symmetric group, determine essentially by the number of incomparable internal nodes in a certain the natural partial order on the tree nodes specified by $p<q$ if p is on the unique path from q to the root. Our bound on the number of classes of merge trees that define the same class of phylogenetic trees is formulated as follows. Recall that we assume that the root of any rooted tree has a unique child.

Proposition A.6. *Let T be a combinatorial phylogenetic tree. Let c denote the unique child of the root vertex. Let A_i be the set of internal nodes of T that are i hops away from c in the path metric (in particular, $A_0=\{c\}$).*

If $\eta(T)$ denotes the number of combinatorial equivalence classes of merge trees indistinguishable from T when regarded as combinatorial phylogenetic trees, then

$$\eta(T) \geq \prod_{j=0}^{k} |A_j|!.$$

Proof. We prove our result by induction on the maximum path distance in T from the child c. If the maximum path distance to the child is 1, then T has a unique internal node c, i.e., T has four nodes: the root r, its child c, and two leaves. This tree admits unique combinatorial merge and phylogenetic structures, whence $\eta(T)=1=1!$.

Suppose now the result holds whenever the maximal path distance from the child c is less than k, for some $k>1$. Decompose the internal nodes of T into k sets $A_1, A_2, \ldots, A_k$. All nodes in A_k have only (two) leaf descendents, as otherwise there would exist an internal node further away from c than some node in A_k, so the maximal path distance to c would be greater than k.

Let $A_k=\{q_1,q_2,\ldots,q_s\}$. If we remove the leaf nodes attached to each $q_i \in A_k$, we obtain a phylogenetic tree T' with internal nodes partitioned into sets $A_1, A_2, \ldots, A_{k-1}$. By the induction hypothesis, there are at least $\prod_{j=1}^{k-1} |A_j|!$ combinatorial equivalence classes of merge trees indistinguishable from T' when considered as phylogenetic trees.

For each such equivalence class, we can obtain merge trees indistinguishable from T as phylogenetic trees by reattaching the leaves to each q_i and choosing any ordering on A_k, which we may do because all q_i are at the same distance from c, and hence are incomparable nodes. Since there are $|A_k|!$ possible total orders on the set of q_i, we can conclude. □

References

[1] Louis J. Billera, Susan P. Holmes, Karen Vogtmann, Geometry of the space of phylogenetic trees, Adv. Appl. Math. 27 (4) (November 2001) 733–767.
[2] Benjamin Brück, Adélie Garin, Stratifying the space of barcodes using Coxeter complexes, J. Appl. Comput. Topol. (12 2022) 1–27.
[3] Michael J. Catanzaro, Justin M. Curry, Brittany Terese Fasy, Jānis Lazovskis, Greg Malen, Hans Riess, Bei Wang, Matthew Zabka, Moduli spaces of Morse functions for persistence, J. Appl. Comput. Topol. 4 (3) (2020) 353–385.
[4] William Crawley-Boevey, Decomposition of pointwise finite-dimensional persistence modules, J. Algebra Appl. 14 (05) (2015) 1550066.
[5] Justin Curry, The fiber of the persistence map for functions on the interval, J. Appl. Comput. Topol. 2 (3) (2018) 301–321.
[6] Justin Curry, Haibin Hang, Washington Mio, Tom Needham, Osman Berat Okutan, Decorated merge trees for persistent topology, arXiv preprint, arXiv: 2103.15804, 2021.
[7] Justin Curry, Sayan Mukherjee, Katharine Turner, How many directions determine a shape and other sufficiency results for two topological transforms, arXiv preprint, arXiv:1805.09782, 2018.
[8] Jacek Cyranka, Konstantin Mischaikow, Charles Weibel, Contractibility of a persistence map preimage, J. Appl. Comput. Topol. 4 (4) (2020) 509–523.
[9] Paul H. Edelman, The Bruhat order of the symmetric group is lexicographically shellable, Proc. Am. Math. Soc. 82 (3) (1981) 355–358.
[10] P. Erdös, Richard K. Guy, J.W. Moon, On refining partitions, J. Lond. Math. Soc. s2–9 (4) (1975) 565–570.
[11] Joseph Felsenstein, The number of evolutionary trees, Syst. Biol. 27 (1) (03 1978) 27–33.
[12] Robin Forman, Morse theory for cell complexes, Adv. Math. 134 (1) (1998) 90–145.
[13] Marcio Gameiro, Yasuaki Hiraoka, Ippei Obayashi, Continuation of point clouds via persistence diagrams, Physica D 334 (2016) 118–132.
[14] Robert Ghrist, Rachel Levanger, Huy Mai, Persistent homology and Euler integral transforms, J. Appl. Comput. Topol. 2 (1) (2018) 55–60.
[15] L. Kanari, H. Dictus, A. Chalimourda, W. Van Geit, B. Coste, J. Shillcock, K. Hess, H. Markram, Computational synthesis of cortical dendritic morphologies, BioArXiv, June 2020, https://www.cell.com/cell-reports/fulltext/S2211-1247(22)00330-8?_returnURL=https%3A%2F%2Flinkinghub.elsevier.com%2Fretrieve%2Fpii%2FS2211124722003308%3Fshowall%3Dtrue.
[16] L. Kanari, P. Dłotko, M. Scolamiero, R. Levi, J. Shillcock, K. Hess, H. Markram, A topological representation of branching neuronal morphologies, Neuroinformatics 16 (1) (Jan 2018) 3–13.
[17] L. Kanari, S. Ramaswamy, Y. Shi, S. Morand, J. Meystre, R. Perin, M. Abdellah, Y. Wang, K. Hess, H. Markram, Objective morphological classification of neocortical pyramidal cells, Cerebral Cortex 29 (2019) 1719–1735.
[18] Lida Kanari, Adélie Garin, Kathryn Hess, From trees to barcodes and back again: theoretical and statistical perspectives, Algorithms 13 (2020).
[19] Kondor Risi, Group theoretical methods in machine learning, PhD thesis, Columbia University, 01 2008.
[20] Michael Lesnick, The theory of the interleaving distance on multidimensional persistence modules, Found. Comput. Math. 15 (3) (2015) 613–650.
[21] Jacob Leygonie, Ulrike Tillmann, The fiber of persistent homology for simplicial complexes, arXiv preprint, arXiv:2104.01372, 2021.
[22] Clément Maria, Steve Oudot, Elchanan Solomon, Intrinsic topological transforms via the distance kernel embedding, arXiv preprint, arXiv:1912.02225, 2019.
[23] D. Morozov, Kenes Beketayev, G. Weber, Interleaving Distance Between Merge Trees, 2013.
[24] Steve Oudot, Elchanan Solomon, Barcode embeddings for metric graphs, arXiv preprint, arXiv:1712.03630, 2017.
[25] Steve Oudot, Elchanan Solomon, Inverse problems in topological persistence, in: Topological Data Analysis, Springer, 2020, pp. 405–433.
[26] Elchanan Solomon, Alexander Wagner, Paul Bendich, From geometry to topology: inverse theorems for distributed persistence, arXiv preprint, arXiv: 2101.12288, 2021.
[27] Chenguang Xu, A correspondence between Schubert cells and persistence diagrams, Master thesis, Supervisor: Yasuaki Hiraoka, Kyoto University, 2020.
[28] Lin Yan, Yusu Wang, Elizabeth Munch, Ellen Gasparovic, Bei Wang, A structural average of labeled merge trees for uncertainty visualization, CoRR, arXiv:1908.00113 [abs], 2019.