Abstract

We study the class of graphs known as $k$-trees through the lens of Joyal’s theory of combinatorial species (and an equivariant extension known as ‘$\Gamma$-species’). This culminates in a system of recursive functional equations giving the generating function for unlabeled $k$-trees which allows for fast, efficient computation of their numbers. Enumerations up to $k = 10$ and $n = 30$ (for a $k$-tree with $n + k - 1$ vertices) are included in tables, and Sage code for the general computation is included in an appendix.

1 Introduction

1.1 $k$-trees

Trees and their generalizations have played an important role in the literature of combinatorial graph theory throughout its history. The multi-dimensional generalization to so-called ‘$k$-trees’ has proved to be particularly fertile ground for both research problems and applications.

The class $\mathfrak{a}_k$ of $k$-trees (for $k \in \mathbb{N}$) may be defined recursively:

**Definition 1.1.** The complete graph on $k$ vertices ($K_k$) is a $k$-tree, and any graph formed by adding a single vertex to a $k$-tree and connecting that vertex by edges to some existing $k$-clique (that is, induced $k$-complete subgraph) of that $k$-tree is a $k$-tree.

The graph-theoretic notion of $k$-trees was first introduced in 1968 in [7]; vertex-labeled $k$-trees were quickly enumerated in the following year in both [10] and [2]. The special case $k = 2$ has been especially thoroughly studied; enumerations are available in the literature for edge- and triangle-labeled 2-trees in [11], for plane 2-trees in [12], and for unlabeled 2-trees in [7] and [6]. In 2001, the theory of species was brought to bear on 2-trees in [4], resulting in more explicit formulas for the enumeration of unlabeled 2-trees. An extensive literature on other properties of $k$-trees and their applications has also emerged; Beineke and Pippert claim in [1] that “[t]here are now over 100 papers on various aspects of $k$-trees”. However, no general enumeration of unlabeled $k$-trees appears in the literature to date.

To begin, we establish two definitions for substructures of $k$-trees which we will use extensively in our analysis.

**Definition 1.2.** A **hedron** of a $k$-tree is a $(k + 1)$-clique and a **front** is a $k$-clique.

We will frequently describe $k$-trees as assemblages of hedra attached along their fronts rather than using explicit graph-theoretic descriptions in terms of edges and vertices, keeping in mind that the structure of interest is graph-theoretic and not geometric. The recursive addition of a single vertex and its connection by edges to an existing $k$-clique in Definition 1.1 is then interpreted as the attachment of a hedron to an existing one along some front, identifying the $k$ vertices they have in common. The analogy to the recursive definition of conventional trees is clear, and in fact the class $\mathfrak{a}$ of trees may be recovered by setting $k = 1$. For higher $k$, the structures formed are still distinctively tree-like; for example, 2-trees are formed by gluing...
triangles together along their edges without forming loops of triangles (see Fig. 1), while 3-trees are formed by gluing tetrahedra together along their triangular faces without forming loops of tetrahedra.

In graph-theoretic contexts, it is conventional to label graphs on their vertices and possibly their edges. However, for our purposes, it will be more convenient to label hedra and fronts. Throughout, we will treat the species \( \mathfrak{a}_k \) of \( k \)-trees as a two-sort species, with \( X \)-labels on the hedra and \( Y \)-labels on their fronts; in diagrams, we will generally use capital letters for the hedron-labels and positive integers for the front-labels (see Fig. 2).

2 The dissymmetry theorem for \( k \)-trees

Studies of tree-like structures—especially those explicitly informed by the theory of species, as ours will be—often feature decompositions based on dissymmetry, which allow enumerations of unrooted structures to be recharacterized in terms of rooted structures. For example, as seen in [3, §4.1], the species \( \mathfrak{a} \) of trees and \( \mathcal{A} = \mathfrak{a}^* \) of rooted trees are related by the equation

\[
\mathcal{A} + \mathcal{E}_2(\mathcal{A}) = \mathfrak{a} + \mathcal{A}^2
\]

where the proof hinges on a recursive structural decomposition of trees. In this case, the species \( \mathcal{A} \) is relatively easy to characterize explicitly, so this equation serves to characterize the species \( \mathfrak{a} \), which would be difficult to do directly. (Readers unfamiliar with the theory of species and its applications to graph enumeration may find a full exposition of the subject in [3].)
A similar theorem holds for $k$-trees.

**Theorem 2.1.** The species $a_k^X$ and $a_k^Y$ of $k$-trees rooted at a hedron and fronts respectively, $a_k^{XY}$ of $k$-trees rooted at a hedron with a designated front, and $a_k$ of unrooted $k$-trees are related by the equation

$$a_k^X + a_k^Y = a_k + a_k^{XY}$$

as an isomorphism of species.

**Proof.** We give a bijective, natural (i.e. label-equivariant) map from $(a_k^X + a_k^Y)$-structures on the left side to $(a_k + a_k^{XY})$-structures on the right side. Define a $k$-path in a $k$-tree to be a non-self-intersecting sequence of consecutively adjacent hedra and fronts, and define the length of a $k$-path to be the total number of hedra and fronts along it. Note that the ends of every maximal $k$-path in a $k$-tree are fronts. It is easily verified, as in [9], that every $k$-tree has a unique center clique (either a hedron or a front) which is the midpoint of every longest $k$-path (or, equivalently, has the greatest $k$-eccentricity, defined appropriately).

An $(a_k^X + a_k^Y)$-structure on the left-hand side of the equation is a $k$-tree $T$ rooted at some clique $c$, which is either a hedron or a front. Suppose that $c$ is the center of $T$. We then map $T$ to its unrooted equivalent in $a_k$ on the right-hand side. This map is a natural bijection from its preimage, the set of $k$-trees rooted at their centers, to $a_k$, the set of unrooted $k$-trees.

Now suppose that the root clique $c$ of the $k$-tree $T$ is not the center, which we denote $C$. Identify the clique $c'$ which is adjacent to $c$ along the $k$-path from $c$ to $C$. We then map the $k$-tree $T$ rooted at the clique $c$ to the same tree $T$ rooted at both $c$ and its neighbor $c'$. This map is also a natural bijection, in this case from the set of $k$-trees rooted at vertices which are not their centers to the set $a_k^{XY}$ of $k$-trees rooted at an adjacent hedron-front pair.

Since these maps are label-equivariant bijections, they induce an isomorphism of species

$$a_k^X + a_k^Y = a_k + a_k^{XY}$$

as desired, completing the proof.

In general we will reformulate the dissymmetry theorem as follows:

**Corollary 2.2.** For the various forms of the species $a_k$ as above, we have

$$a_k = a_k^X + a_k^Y - a_k^{XY}.$$  

as an isomorphism of species.

This species subtraction is well-defined in the sense that since the species $a_k^{XY}$ embeds in the species $a_k^X + a_k^Y$ by the centering map described in the proof of Theorem 2.1. Essentially, Eq. (2) identifies each unrooted $k$-tree with itself rooted at its center simplex. This may be understood formally either in the sense of virtual species as in [3, §2.5] or in the sense of species maps as in [5, Def. 1.3.3].

Theorem 2.1 and the consequent Eq. (2) allow us to reframe enumerative questions about generic $k$-trees in terms of questions about $k$-trees rooted in various ways. However, the rich internal symmetries of large cliques obstruct direct analysis of these rooted structures. We need to break these symmetries to proceed.

### 3 Coherently-oriented $k$-trees

#### 3.1 Symmetry-breaking

In the case of the species $A = a_{1}^{\cdot}$ of rooted trees, we may obtain a simple recursive functional equation [3, §1, eq. (9)]:

$$A = X \cdot E(A).$$

This completely characterizes the combinatorial structure of the class of trees.

However, in the more general case of $k$-trees, no such simple relationship obtains; attached to a given hedron is a collection of sets of hedra (one such set per front), but simply specifying which fronts to attach to which does not fully specify the attachings, and the structure of that collection of sets is complex. We will break this symmetry by adding additional structure which we can later remove using the theory of quotient species.
**Definition 3.1.** Let \( h_1 \) and \( h_2 \) be two hedra joined at a front \( f \), hereafter said to be adjacent. Each other front of one of the hedra shares \( k-1 \) vertices with \( f \); we say that two fronts \( f_1 \) of \( h_1 \) and \( f_2 \) of \( h_2 \) are mirror with respect to \( f \) if these shared vertices are the same, or equivalently if \( f_1 \cap f = f_2 \cap f \).

**Observation 3.2.** Let \( T \) be a \( k \)-tree with two hedra \( h_1 \) and \( h_2 \) joined at a front \( f \). Then there is exactly one front of \( h_2 \) mirror to each front of \( h_1 \) with respect to their shared front \( f \).

**Definition 3.3.** Define an orientation of a hedron to be a cyclic ordering of the set of its fronts and an orientation of a \( k \)-tree to be a choice of orientation for each of its hedra. If two oriented hedra share a front, their orientations are compatible if they correspond under the mirror bijection. Then an orientation of a \( k \)-tree is coherent if every pair of adjacent hedra is compatibly-oriented.

See Fig. 3 for an example. Note that every \( k \)-tree admits many coherent orientations—any one hedron of the \( k \)-tree may be oriented freely, and a unique orientation of the whole \( k \)-tree will result from each choice of such an orientation of one hedron. We will denote by \( \mathbf{a}_k \) the species of coherently-oriented \( k \)-trees.

By shifting from the general \( k \)-tree setting to that of coherently-oriented \( k \)-trees, we break the symmetry described above. If we can now establish a group action on \( \mathbf{a}_k \) whose orbits are generic \( k \)-trees we can use the theory of quotient species to extract the generic species \( \mathbf{a}_k \). First, however, we describe an encoding procedure which will make future work more convenient.

**3.2 Bicolored tree encoding**

Although \( k \)-trees are graphs (and hence made up simply of edges and vertices), their structure is more conveniently described in terms of their simplicial structure of hedra and fronts. Indeed, if each hedron has an orientation of its faces and we choose in advance which hedron to attach to which by what fronts, the requirement that the resulting \( k \)-tree be coherently oriented is strong enough to characterize the attaching completely. We thus pass from coherently-oriented \( k \)-trees to a surrogate structure which exposes the salient features of this attaching structure more clearly—structured bicolored trees in the spirit of the \( R, S \)-enriched bicolored trees of [3, §3.2].

A \((\mathcal{E}_{k+1}, \mathcal{E})\)-enriched bicolored tree is a bicolored tree each black vertex of which carries a \( \mathcal{E}_{k+1} \)-structure (that is, a cyclic ordering on \( k+1 \) elements) on its white neighbors. (The \( \mathcal{E} \)-structure on the black neighbors of each white vertex is already implicit in the bicolored tree itself.) For later convenience, we will sometimes call such objects \( k \)-coding trees, and we will denote by \( \mathcal{C}T_k \) the species of such \( k \)-coding trees.

We now define a map \( \beta : \mathbf{a}_k[n] \to \mathcal{C}T_k[n] \). For a given coherently-oriented \( k \)-tree \( T \) with \( n \) hedra:

- For every hedron of \( T \) construct a black vertex and for every front a white vertex, assigning labels appropriately.
• For every black-white vertex pair, construct a connecting edge if the white vertex represents a front of the hedron represented by the black vertex.

• Finally, enrich the collection of neighbors of each black vertex with a $C_{k+1}$-structure inherited directly from the orientation of the $k$-tree $T$.

The resulting object $\beta(T)$ is clearly a $k$-coding tree with $n$ black vertices.

We can recover a $T$ from $\beta(T)$ by following the reverse procedure. For an example, see Fig. 4, which shows the 2-coding tree associated to the coherently-oriented 2-tree of Fig. 3. Note that, for clarity, we have rendered the black vertices (corresponding to hedra) with squares.

Figure 4: A $(C_{k+1}, E)$-enriched bicolored tree encoding a coherently-oriented 2-tree

**Theorem 3.4.** The map $\beta$ induces an isomorphism of species $\vec{a}_k \simeq C T_k$.

**Proof.** It is clear that $\beta$ sends each coherently-oriented $k$-tree to a unique $k$-coding tree, and that this map commutes with permutations on the label sets (and thus is categorically natural). To show that $\beta$ induces a species isomorphism, then, we need only show that $\beta$ is a surjection onto $CT_k[n]$ for each $n$. Throughout, we will say ‘$F$ and $G$ have contact of order $n$’ when the restrictions $F_{\leq n}$ and $G_{\leq n}$ of the species $F$ and $G$ to label sets of cardinality at most $n$ are naturally isomorphic.

First, we note that there are exactly $k!$ coherently-oriented $k$-trees with one hedron—one for each cyclic ordering of the $k + 1$ front labels. There are also $k!$ coding trees with one black vertex, and the encoding $\beta$ is clearly a natural bijection between these two sets. Thus, the species $\vec{a}_k$ of coherently-oriented $k$-trees and $CT_k$ of $k$-coding trees have contact of order 1.

Now, by way of induction, suppose $\vec{a}_k$ and $CT_k$ have contact of order $n \geq 1$. Let $C$ be a $k$-coding tree with $n + 1$ black vertices. Then let $C_1$ and $C_2$ be two distinct sub-$k$-coding trees of $C$, each obtained from $C$ by removing one black node which has only one white neighbor which is not a leaf. Then, by hypothesis, there exist coherently-oriented $k$-trees $T_1$ and $T_2$ with $n$ hedra such that $\beta(T_1) = C_1$ and $\beta(T_2) = C_2$. Moreover, $\beta(T_1 \cap T_2) = \beta(T_1) \cap \beta(T_2)$, and this $k$-coding tree has $n - 1$ black vertices, so $T_1 \cap T_2$ has $n - 1$ hedra. Thus, $T = T_1 \cup T_2$ is a coherently-oriented $k$-tree with $n + 1$ black hedra, and $\beta(T) = C$ as desired. Thus, $\beta^{-1}(\beta(T_1) \cup \beta(T_2)) = T_1 \cup T_2 = T$, and hence $\vec{a}_k$ and $CT_k$ have contact of order $n + 1$.

Thus, $\vec{a}_k$ and $CT_k$ are isomorphic as species; however, $k$-coding trees are much simpler than coherently-oriented $k$-trees as graphs. Moreover, $k$-coding trees are doubly-enriched bicolored trees as in [3, §3.2], for which the authors of that text develop a system of functional equations which fully characterizes the cycle
index of such a species. We thus will proceed in the following sections with a study of the species $\mathcal{C}T_k$, then lift our results to the $k$-tree context.

### 3.3 Functional decomposition of $k$-coding trees

With the encoding $\beta : \tilde{a}_k \rightarrow \mathcal{C}T_k$, we now have direct graph-theoretic access to the attaching structure of coherently-oriented $k$-trees. We therefore turn our attention to the $k$-coding trees themselves to produce a recursive decomposition. As with $k$-trees, we will study rooted versions of the species $\mathcal{C}T_k$ of $k$-coding trees first, then use dissymmetry to apply the results to unrooted enumeration.

**Theorem 3.5.** The species $\mathcal{C}T^X_k$ of $X$-rooted $k$-coding trees, $\mathcal{C}T^Y_k$ of $Y$-rooted $k$-coding trees, and $\mathcal{C}T^{XY}_k$ of edge-rooted $k$-coding trees satisfy the functional equations

\begin{align}
\mathcal{C}T^X_k &= X \cdot e_{k+1}(\mathcal{C}T^Y_k) \\
\mathcal{C}T^Y_k &= Y \cdot e \left( X \cdot L_k(\mathcal{C}T^Y_k) \right) \\
\mathcal{C}T^{XY}_k &= \mathcal{C}T^Y_k \cdot X \cdot L_k(\mathcal{C}T^Y_k) = X \cdot L_{k+1}(\mathcal{C}T^Y_k)
\end{align}

as isomorphisms of species.

**Proof.** By construction, a $\mathcal{C}T^X_k$-structure consists of a single $X$-label and a cyclically-ordered $(k + 1)$-set of $\mathcal{C}T^Y_k$-structures. This gives Eq. (4a). See Fig. 5 for an example of this construction.

![Figure 5: An example $\mathcal{T}_4^X$-structure, rooted at the $X$-vertex.](image)

Similarly, a $\mathcal{C}T^Y_k$-structure consists of a single $Y$-label and a (possibly empty) set of structures which are a slight variant of the $\mathcal{T}_4^X$-structures discussed above. Every white neighbor of the black root of a $\mathcal{T}_4^X$-structure is labeled in the construction above, but the white parent of a $\mathcal{T}_4^X$-structure in this recursive decomposition is already labeled. Thus, the structure around a black vertex which is a child of a white vertex consists of an $X$ label and a linearly-ordered $k$-set of $\mathcal{T}_4^Y$-structures. Thus, a $\mathcal{T}_4^Y$-structure consists
of a $Y$-label and a set of pairs of an $X$ label and an $\mathcal{L}_k$-structure of $\mathcal{C}T_k^Y$-structures. This gives Eq. (4b). We note here for conceptual consistency that in fact $\mathcal{L}_k = \mathcal{C}'_{k+1}$ for $\mathcal{L}$ the species of linear orders and $\mathcal{C}$ the species of cyclic orders and that $\mathcal{E}' = \mathcal{E}$ for $\mathcal{E}$ the species of sets; readers familiar with the $R, S$-enriched bicolored trees of [3, §3.2] will recognize echoes of their decomposition in these facts.

Finally, a $\mathcal{C}T_k^{XY}$-structure is simply an $X \cdot \mathcal{L}_k(\mathcal{C}T_k^Y)$-structure as described above (corresponding to the black vertex) together with a $\mathcal{C}T_k^Y$-structure (corresponding to the white vertex). For reasons that will become clear later, we note that we can incorporate the root white vertex into the linear order by making it last, thus representing a $\mathcal{C}T_k^{XY}$-structure instead as an $X \cdot \mathcal{L}_{k+1}(\mathcal{C}T_k^Y)$-structure. This gives Eq. (4c). See Fig. 6 for an example of this construction.

![Figure 6: An example $\mathcal{C}T_k^{XY}$-structure, rooted at the X-vertex and the thick edge adjoining it.](image)

However, a recursive characterization of the various species of $k$-coding trees is insufficient to characterize the species of $k$-trees itself, since $k$-coding trees represent $k$-trees with coherent orientations.

### 4 Generic $k$-trees

In [4], the orientation-reversing action of $S_2$ on $\text{Cyc}_3$ is exploited to study 2-trees species-theoretically. We might hope to develop an analogous group action under which general $k$-trees are naturally identified with orbits of coherently-oriented $k$-trees under an action of $S_k$. Unfortunately:

**Proposition 4.1.** For $k \geq 3$, no transitive action of any group on the set $\text{Cyc}_{[k+1]}$ of cyclic orders on $[k+1]$ commutes with the action of $S_{k+1}$ that permutes labels.

**Proof.** We represent the elements of $\text{Cyc}_{[k+1]}$ as cyclic permutations on the alphabet $[k+1]$; then the action of $S_{k+1}$ that permutes labels is exactly the conjugation action on these permutations. Consider an action of a group $G$ on $\text{Cyc}_{[k+1]}$ that commutes with this conjugation action. Then, for any $g \in G$ and any $c \in \text{Cyc}_{[k+1]}$, we have that

$$g \cdot c = g \cdot c c^{-1} = c(g \cdot c)c^{-1}$$

(5)

7
and so \( c \) and \( g \cdot c \) commute. Thus, \( c \) commutes with every element of its orbit under the action of \( G \). But, for \( k \geq 3 \), not all elements of \( \text{Cyc}_{[k+1]} \) commute, so the action is not transitive.

We thus cannot hope to attack the coherent orientations of \( k \)-trees by acting directly on the cyclic orderings of fronts. Instead, we will use the additional structure on rooted coherently-oriented \( k \)-trees; with rooting, the cyclic orders around black vertices are converted into linear orders, for which there is a natural action of \( \mathcal{S}_{k+1} \).

### 4.1 Group actions on \( k \)-coding trees

We have noted previously that every labeled \( k \)-tree admits exactly \( k! \) coherent orientations. Thus, there are \( k! \) distinct \( k \)-coding trees associated to each labeled \( k \)-tree, which differ only in the \( \mathcal{C}_{k+1} \)-structures on their black vertices. Consider a rooted \( k \)-coding tree \( T \) and a black vertex \( v \) which is not the root vertex. Then one white neighbor of \( v \) is the ‘parent’ of \( v \) (in the sense that it lies on the path from \( v \) to the root). We thus can convert the cyclic order on the \( k + 1 \) white neighbors of \( v \) to a linear order by choosing the parent white neighbor to be last. There is a natural, transitive, label-independent action of \( \mathcal{S}_{k+1} \) on the set of such linear orders which induces an action on the cyclic orders from which the linear orders are derived. However, only elements of \( \mathcal{S}_{k+1} \) which fix \( k + 1 \) will respect the structure around the black vertex we have chosen, since its parent white vertex must remain last.

In addition, if we simply apply the action of some \( \sigma \in \mathcal{S}_{k+1} \) to the order on white neighbors of \( v \), we change the coherently-oriented \( k \)-tree \( \beta^{-1}(T) \) to which \( T \) is associated in such a way that it no longer corresponds to the same unoriented \( k \)-tree. Let \( t \) denote the unoriented \( k \)-tree associated to \( \beta^{-1}(T) \); then there exists a coherent orientation of \( t \) which agrees with orientation around \( v \) induced by \( \sigma \). The \( k \)-coding tree \( T' \) corresponding to this new coherent orientation has the same underlying bicolored tree as \( T \) but possibly different orders around its black vertices. If we think of the \( k \)-coding tree \( T' \) as the image of \( T \) under a global action of \( \sigma \), orbits under all of \( \mathcal{S} \) will be precisely the classes of \( k \)-coding trees corresponding to all coherent orientations of specified \( k \)-trees, allowing us to study unoriented \( k \)-trees as quotients. The orientation of \( T' \) will be that obtained by applying \( \sigma \) at \( v \) and then recursively adjusting the other cyclic orders so that fronts which were mirror are made mirror again. This will ensure that the combinatorial structure of the underlying \( k \)-tree \( t \) is preserved.

Therefore, when we apply some permutation \( \sigma \in \mathcal{S}_{k+1} \) to the white neighbors of a black vertex \( v \), we must also permute the cyclic orders of the descendant black vertices of \( v \). In particular, the permutation \( \sigma' \) which must be applied to some immediate black descendant \( v' \) of \( v \) is precisely the permutation on the linear order of white neighbors of \( v' \) induced by passing over the mirror bijection from \( v' \) to \( v \), applying \( \sigma \), and then passing back. We can express this procedure in formulaic terms:

**Theorem 4.2.** If a permutation \( \sigma \in \mathcal{S}_{k+1} \) is applied to a linearized orientation of a black vertex \( v \) in rooted \( k \)-coding tree, the permutation which must be applied to the linearized orientation a child black vertex \( v' \) which was attached to the \( i \)th white child of \( v \) (with respect to the linear ordering induced by the orientation) to preserve the mirror relation is \( \rho_i(\sigma) \), where \( \rho_i \) is the map given by

\[
\rho_i(\sigma) : a \mapsto \sigma(i + a) - \sigma(i) \tag{6}
\]

in which all sums and differences are reduced to their representatives modulo \( k + 1 \) in \( \{1, 2, \ldots, k + 1\} \).

**Proof.** Let \( v' \) denote a black vertex which is attached to \( v \) by the white vertex 1, which we suppose to be in position \( i \) in the linear order induced by the original orientation of \( v \). Let 2 denote the white neighbor of \( v' \) which is \( a \)th in the linear order induced by the original orientation around \( v' \). It is mirror to the white neighbor 3 of \( v \) which is \( (i + a) \)th in the linear order induced by the original orientation around \( v \). After the action of \( \sigma \) is applied, vertex 3 is \( \sigma(i + a) \)th in the new linear order around \( v \). We require that 2 is still mirror to 3, so we must move it to position \( \sigma(i + a) - \sigma(i) \) when we create a new linear order around \( v' \). This completes the proof.

This procedure is depicted in Fig. 7.

As an aside, we note that, although the construction \( \rho \) depends on \( k \), the value of \( k \) will be fixed in any given context, so we suppress it in the notation.
Any $\sigma$ which is to be applied to a non-root black vertex $v$ must of course fix $k+1$. We can think of $\mathfrak{S}_k$ as the subgroup of $\mathfrak{S}_{k+1}$ of permutations fixing $k+1$, and so in what follows we will refer to this as a $\mathfrak{S}_k$-action where appropriate.

In light of Theorem 3.5, we now wish to adapt these ideas into explicit $\mathfrak{S}_k$- and $\mathfrak{S}_{k+1}$-actions on $\mathcal{CT}_k^Y$, $\mathcal{CT}_k^{XY}$, and $\mathcal{CT}_k^{YX}$ whose orbits correspond to the various coherent orientations of single underlying rooted $k$-trees. In the case of a $Y$-rooted $k$-coding tree $T$, if we declare that $\sigma \in \mathfrak{S}_k$ acts on $T$ by acting directly on each of the black vertices immediately adjacent to the root and then applying $\rho$-derived permutations recursively to their descendants, orbits behave as expected. The same $\mathfrak{S}_k$-action serves equally well for edge-rooted $k$-coding trees, where (for purposes of applying the action of some $\sigma$) we can simply ignore the black vertex in the root.

However, if we begin with an $X$-rooted $k$-coding tree, the cyclic ordering of the white neighbors of the root black vertex has no canonical choice of linearization. If we make an arbitrary choice of one of the $k+1$ available linearizations, and thus convert to an edge-rooted $k$-coding tree, the full $\mathfrak{S}_{k+1}$-action defined previously can be applied directly to the root vertex. The orbit under this action of some edge-rooted $k$-coding tree $T$ with a choice of linearization at the root then includes all possible linearizations of the root orders of all possible $X$-rooted $k$-coding trees corresponding to the different coherent orientations of a single $k$-coding tree.

It follows that:

**Lemma 4.3.** *The actions of $\mathfrak{S}_k$ on $\mathcal{CT}_k^Y$ and $\mathcal{CT}_k^{XY}$ and the action of $\mathfrak{S}_{k+1}$ on $\mathcal{CT}_k^{YX}$ are transitive in the sense that each orbit corresponds to the set of all coherent orientations of a single underlying rooted unoriented $k$-tree. Moreover, these actions all commute with the $\mathfrak{S}_n$-actions which permute labels.***

### 4.2 $k$-trees as quotients

We have now equipped the various species of rooted $k$-coding trees with actions which commute with permutations of labels, making them ‘species-compatible’. Moreover, the non-oriented rooted $k$-trees underlying the $k$-coding trees are naturally identified with orbits under these actions, suggesting that rooted $k$-trees are ‘quotients’ of rooted $k$-coding trees. The notion of a $\Gamma$-species as developed in [8, §3] formalizes this notion of compatibility and provides an enumerative toolset for dealing with quotients of this sort. In this language, we will treat $\mathcal{CT}_k^Y$ and $\mathcal{CT}_k^{XY}$ as $\mathfrak{S}_k$-species and $\mathcal{CT}_k^{YX}$ as an $\mathfrak{S}_{k+1}$-species$^1$ (indicating that they carry equivariant actions of the specified group). (Hereafter, when it is necessary to distinguish, a species which is not equipped with any $\Gamma$-species structure will be dubbed an ‘actionless’ species.)

As a consequence of Lemma 4.3, then, we can then relate the rooted $\Gamma$-species forms of $\mathcal{CT}_k$ to the various (actionless) species forms of generic rooted $k$-trees in Theorem 2.1:

---

$^1$The $\mathfrak{S}_k$- and $\mathfrak{S}_{k+1}$-actions on $\mathcal{CT}_k^{XY}$ are compatible, but we will make explicit reference to $\mathcal{CT}_k^{XY}$ as an $\mathfrak{S}_k$- or $\mathfrak{S}_{k+1}$-species whenever it is important and not completely clear from context which we mean.
Theorem 4.4. For the various rooted forms of the (actionless) species $a_k$ as in Theorem 2.1 and the various rooted $\Gamma$-species forms of $CT_k$ as in Theorem 3.5 (interpreted as $S_k$- and $S_{k+1}$-species), we have

\begin{align}
  a_k^Y &= \frac{CT_k^Y}{S_k} \\
  a_k^{XY} &= \frac{CT_k^{XY}}{S_k} \\
  a_k^X &= \frac{CT_k^{XY}}{S_{k+1}}
\end{align}

as isomorphisms of (actionless) species, where $CT_k^{XY}$ is an $S_k$-species in Eq. (7a) and an $S_{k+1}$-species in Eq. (7c).

As a result, we have explicit characterizations of all the rooted components of the original dissymmetry theorem, Theorem 2.1. Thus, through the enumerative toolset of species theory, we can enumerate $k$-trees through a careful enumeration of each of $CT_k^Y$ and $CT_k^{XY}$.

5 Automorphisms and cycle indices

Species theory associates to each species $F$ an enumerative power series $Z_F$ dubbed the ‘cycle index’ which keeps track of the number of structures with a given automorphism type. $\Gamma$-species theory provides a natural extension of the cycle index for a $\Gamma$-species $F$, denoted $Z_{\Gamma}F$, which keeps track of the number of $\gamma$-invariant structures with a given automorphism type for each $\gamma \in \Gamma$, defined in [8, §3]. We reprint its definition here for convenience:

\[
Z_{\Gamma}F(\gamma) := \frac{1}{|\Gamma|} \sum_{n \geq 0} \frac{1}{n!} \sum_{\sigma \in S_n} \text{fix}(\gamma \cdot F[\sigma]) p_\sigma,
\]

where $p_\sigma = p_{\sigma_1}^1 p_{\sigma_2}^2 \ldots$ (for $\sigma_i$ the number of $i$-cycles of $\sigma$) is the monomial recording the cycle structure of $\sigma$.

Just as with classical cycle indices (or enumerative generating functions of non-species-theoretic combinatorics), the algebra of $\Gamma$-cycle indices is closely associated to the combinatorial algebra of their species. We will now apply the results of the preceding sections to compute the cycle indices of the various $\Gamma$-species we have developed.

5.1 $k$-coding trees: $CT_k^Y$ and $CT_k^{XY}$

Corollary 2.2 of the dissymmetry theorem for $k$-trees has a direct analogue in terms of cycle indices:

Theorem 5.1. For the various forms of the species $a_k$ as in Section 2, we have

\[Z_{a_k} = Z_{a_k^X} + Z_{a_k^Y} - Z_{a_k^{XY}}.\]

Thus, we need to calculate the cycle indices of the three rooted forms of $a_k$. A straightforward application of Burnside’s lemma allows us to pass from the cycle index of a $\Gamma$-species $F$ to the ordinary cycle index of the quotient species $\Gamma/F$:

Lemma 5.2. For a $\Gamma$-species $F$, the ordinary cycle index of the quotient species $F/\Gamma$ is given by

\[Z_{F/\Gamma} = \overline{Z_F} := \frac{1}{|\Gamma|} \sum_{\gamma \in \Gamma} Z_F^{\Gamma}(\gamma) = \frac{1}{|\Gamma|} \sum_{n \geq 0} \frac{1}{n!} (\gamma \cdot F[\sigma]) p_\sigma,
\]

where we define $\overline{Z_F} = \frac{1}{|\Gamma|} \sum_{\gamma \in \Gamma} Z_F^{\Gamma}(\gamma)$ for future convenience.

From Theorem 4.4 and by Lemma 5.2 we obtain:
Theorem 5.3. For the various forms of the species $a_k$ as in Section 2 and the various $S_k$-species and $S_{k+1}$-species forms of $\mathcal{C}T_k$ as in Section 4.1, we have

\[
Z_{a_k}^Y = \frac{Z^{S_k}_{\mathcal{C}T_k^Y}}{k!} = \frac{1}{k!} \cdot \sum_{\sigma \in S_k} Z^{S_k}_{\mathcal{C}T_k^Y} (\sigma) \tag{9a}
\]

\[
Z_{a_k}^X = \frac{Z^{S_k}_{\mathcal{C}T_k^XY}}{k!} = \frac{1}{k!} \cdot \sum_{\sigma \in S_k} Z^{S_k}_{\mathcal{C}T_k^XY} (\sigma) \tag{9b}
\]

\[
Z_{a_k}^X = \frac{Z^{S_{k+1}}_{\mathcal{C}T_k^{XY}}}{(k+1)!} = \frac{1}{(k+1)!} \cdot \sum_{\sigma \in S_{k+1}} Z^{S_{k+1}}_{\mathcal{C}T_k^{XY}} (\sigma) \tag{9c}
\]

We thus need only calculate the various $\Gamma$-cycle indices for the $S_k$-species and $S_{k+1}$-species forms of $\mathcal{C}T_k^Y$ and $\mathcal{C}T_k^{XY}$ to complete our enumeration of general $k$-trees.

In Theorem 5.3, the functional equations for the (actionless) species $\mathcal{C}T_k^Y$ and $\mathcal{C}T_k^{XY}$ both include terms of the form $L_k \circ \mathcal{C}T_k^Y$. The plethysm of (actionless) species does have a generalization to $\Gamma$-species, as given in [8, §3], but it does not correctly describe the manner in which $\mathcal{S}_k$ acts on linear orders of $\mathcal{C}T_k^Y$-structures in these recursive decompositions. Specifically, for two $\Gamma$-species $F$ and $G$, an element $\gamma \in \Gamma$ acts on an $(F \circ G)$-structure (colloquially, 'an $F$-structure of $G$-structures') by acting on the $F$-structure and on each of the $G$-structures independently. In our action of $\mathcal{S}_k$, however, the actions of $\sigma$ on the descendant $\mathcal{C}T_k^Y$-structures are not independent—they depend on the position of the structure in the linear ordering around the parent black vertex. In particular, if $\sigma$ acts on some non-root black vertex, then $\rho_i(\sigma)$ acts on the white vertex in the $i$th place, where in general $\rho_i(\sigma) \neq \sigma$.

Thus, we consider automorphisms of these $S_k$-structures directly. First, we consider the component species $X \cdot L_k(\mathcal{C}T_k^Y)$.

Lemma 5.4. Let $B$ be a structure of the species $X \cdot L_k(\mathcal{C}T_k^Y)$. Let $W_i$ be the $\mathcal{C}T_k^Y$-structure in the $i$th position in the linear order. Then some $\sigma \in \mathcal{S}_k$ acts as an automorphism of $B$ if and only if, for each $i \in [k+1]$, we have $(\rho_i(\sigma) W_i \cong W_{\sigma(i)}$.

Proof. Recall that the action of $\sigma \in \mathcal{S}_k$ is in fact the action of the lift of $\sigma$ as an element of $\mathcal{S}_{k+1}$. The $X$-label on the black root of $B$ is not affected by the action of $\sigma$, so no conditions on $\sigma$ are necessary to accommodate it. However, the $L_k$-structure on the white children of the root is permuted by $\sigma$, and we apply to each of the $W_i$’s the action of $(\rho_i(\sigma)$. Thus, $\sigma$ is an automorphism of $B$ if and only if the combination of applying $\sigma$ to the linear order and $\rho_i(\sigma$ to each $W_i$ is an automorphism. Since $\sigma$ ‘carries’ each $W_i$ onto $W_{\sigma(i)}$, we must have that $(\rho_i(\sigma) W_i \cong W_{\sigma(i)}$, as claimed. That this suffices is clear. \qed

We now have the tools in hand to find recursive functional equations satisfied by $Z^{S_k}_{\mathcal{C}T_k^Y}$ and $Z^{S_k}_{\mathcal{C}T_k^{XY}}$.

Theorem 5.5 (Cycle index of $\mathcal{C}T_k^Y$). The $\mathcal{S}_k$-cycle index for the species $\mathcal{C}T_k^Y$ is characterized by the recursive functional equation

\[
Z^{S_k}_{\mathcal{C}T_k^Y} (\sigma) = p_1 | y | \cdot Z_c \circ \left( p_1 | x | \cdot \prod_{c \in C(\sigma)} Z^{S_k}_{\mathcal{C}T_k^Y} (\prod_{i \in c} \rho_i(\sigma)) \left( p_{|c|}|x|, p_{2|c|}|x|, \ldots ; p_{|c|}|y|, p_{2|c|}|y|, \ldots \right) \right) \tag{10}
\]

where $C(\sigma)$ denotes the set of cycles of $\sigma$ (as a $k$-permutation), where the inner product is taken in order with respect to any choice of linearization of the cyclic order of the elements of $c$, and where the plethysm $\circ$ is that of ordinary cycle indices.

Proof. Let $p_\pi | x |$ and $p_\tau | y |$ be the monomials in $x$- and $y$-variables recording the cycle structure of $\sigma$, analogously to $p_\sigma | x |$ in the explanation of Eq. (8). Then, by definition, for permutations $\pi$ acting on black vertices and $\tau$ on white, the coefficient on the $p_\pi | x | p_\tau | y |$-term of $Z^{S_k}_{\mathcal{C}T_k^Y} (\sigma)$ is $\frac{1}{k!}$ times the number of $\mathcal{C}T_k^Y$-structures for which $\sigma \cdot \mathcal{C}T_k^Y[\pi, \tau]$ is an automorphism. However, the actions of $\sigma$ and of $\pi$ and $\tau$ commute, so this is equal to $\frac{1}{k!}$ times the number of $\sigma$-invariant $\mathcal{C}T_k^Y$-structures of automorphism type $[\pi, \tau]$.

For each $\sigma \in \mathcal{S}_k$, let $\mathcal{C}T_k^Y(\sigma)$ be the ordinary two-sort species of $\sigma$-invariant $Y$-rooted $k$-coding trees. Then we conclude that

\[
Z^{S_k}_{\mathcal{C}T_k^Y} (\sigma) = Z_{\mathcal{C}T_k^Y}(\sigma), \tag{11}
\]
Now consider a structure $T$ of the species $\mathcal{C}^Y \mathcal{T}_k^Y(\sigma)$—that is, a $\sigma$-invariant $\mathcal{C}^Y \mathcal{T}_k^Y$-structure. We concluded in Theorem 3.5 that $\mathcal{C}^Y \mathcal{T}_k^Y = Y \cdot \mathcal{E}(X \cdot \mathcal{L}_k(\mathcal{C}^Y \mathcal{T}_k^Y))$; we now consider the effect that the requirement of $\sigma$-invariance has on this functional relationship. We will work at the cycle-index level for convenience, since the effect on the $\mathcal{L}_k$-component does not correspond to any conventional species-algebra operation.

As a $\mathcal{C}^Y \mathcal{T}_k^Y$-structure, $T$ must have a $Y$-labeled root which is fixed by any automorphism of $T$. The root therefore contributes a $p_1[y]$ term to $Z_{\mathcal{C}^Y \mathcal{T}_k^Y(\sigma)}$, just as it did to $Z_{\mathcal{C}^Y \mathcal{T}_k^Y}$.

The $Y$-labeled root of $T$ is adjoined by some set of $X$-labeled black neighbors. As a $\mathcal{C}^Y \mathcal{T}_k^Y$-structure, $T$ already carries an $\mathcal{E}$-structure on these black neighbors of the root. This $\mathcal{E}$-structure is unaffected by the action of $\sigma$. In Theorem 3.5, we found that each of these $X$-rooted descendants carried a $(X \cdot \mathcal{L}_k(\mathcal{C}^Y \mathcal{T}_k^Y))$-structure. Let $F$ denote the ordinary two-sort species of $\sigma$-invariant $(X \cdot \mathcal{L}_k(\mathcal{C}^Y \mathcal{T}_k^Y))$-structures; then

$$Z_{\mathcal{C}^Y \mathcal{T}_k^Y(\sigma)} = p_1[y] \cdot (Z_E \circ Z_F) \tag{12}$$

(where the plithysm is that of ordinary cycle indices). It remains only to compute $Z_F$.

Let $T'$ be a structure of the ordinary two-sort species $F$—that is, a $\sigma$-invariant descendant $X$-rooted $k$-coding tree (i.e., one with only an $\mathcal{L}_k$-structure of white neighbors instead of an $\mathcal{L}_{k+1}$-structure). As before, the requirement of $\sigma$-invariance has no effect on the root, so there is still a contribution of $p_1[x]$ to the cycle index $Z_F$. However, by construction, $\sigma$ changes the $\mathcal{L}_k$-structure of the descendants of the root of $T'$.

Since $T'$ is $\sigma$-invariant, by Lemma 5.4, we must have that $\sigma$ carries the $\mathcal{C}^Y \mathcal{T}_k^Y$-structure at $i$ to its $\rho_i(\sigma)$-image. Thus, for each cycle of $\sigma$, only one $\mathcal{C}^Y \mathcal{T}_k^Y$-structure may be chosen, after which all its images under $\sigma$ are determined. Furthermore, if the cycle $c$ containing $i$ is of length $|c|$, then the $\mathcal{C}^Y \mathcal{T}_k^Y$-structure at $i$ must be carried back onto itself by $|c|$ successive applications of $\sigma$. Thus, the structure at $i$ must be $(\prod_{i \in c} \rho_i(\sigma))$-invariant. Since there are $|c|$ copies of this $\mathcal{C}^Y \mathcal{T}_k^Y$-structure, the contribution to $Z_F$ of each cycle $c$ of $\sigma$ is

$$Z_{\mathcal{C}^Y \mathcal{T}_k^Y(\sigma)} = p_1[y] \cdot Z_E \circ \left( p_1[x] \cdot \prod_{c \in \mathcal{C}(\sigma)} Z_{\mathcal{C}^Y \mathcal{T}_k^Y} \left( \prod_{i \in c} \rho_i(\sigma) \right) \right) \tag{13}$$

Combining Eqs. (12) and (13) we conclude that

$$Z_{\mathcal{C}^Y \mathcal{T}_k^Y(\sigma)} = p_1[y] \cdot Z_E \circ \left( p_1[x] \cdot \prod_{c \in \mathcal{C}(\sigma)} Z_{\mathcal{C}^Y \mathcal{T}_k^Y} \left( \prod_{i \in c} \rho_i(\sigma) \right) \right)$$

and so Eq. (11) completes the proof of Eq. (10).

**Theorem 5.6** (Cycle index of $\mathcal{C}^{XY} \mathcal{T}_k^X$). The $\mathcal{S}_{k+1}$-cycle index for the species $\mathcal{C}^Y \mathcal{T}_k^X$ is given by

$$Z_{\mathcal{C}^{XY} \mathcal{T}_k^X(\sigma)} = p_1[x] \cdot \prod_{c \in \mathcal{C}(\sigma)} Z_{\mathcal{C}^Y \mathcal{T}_k^X} \left( \prod_{i \in c} \rho_i(\sigma) \right) \tag{14}$$

under the same conditions as Theorem 5.5.

**Proof.** The proof is essentially identical to that of Theorem 5.5.

Terms of the form $\prod_{i \in c} \rho_i(\sigma)$ appear in Eqs. (10) and (14). For the simplification of calculations, we note here two useful results about these products.

First, we observe that certain $\rho$-maps preserve cycle structure:

**Lemma 5.7.** Let $\sigma \in \mathcal{S}_k$ be a permutation of which $i \in [k]$ is a fixed point. Then $\rho_i(\sigma)$ has the same cycle type as $\sigma$. 

12
Proof. Let $\theta_i \in S_k$ denote the permutation given by $\theta_i(a) = a + i$ reduced modulo $k + 1$ as in the proof of Theorem 4.2. Then, since $i$ is a fixed point of $\sigma$, we have that, for each $a \in \{1, 2, \ldots, k + 1\}$,

$$\rho_i(\sigma)(a) = \sigma(i + a) - i = (\theta_i \sigma \theta_i^{-1})(a),$$

so $\rho_i(\sigma)$ is a conjugate of $\sigma$ and thus has the same cycle structure as $\sigma$, as desired. \hfill \Box

But then we note that the products in the above theorems are in fact permutations obtained by applying such $\rho$-maps:

**Lemma 5.8.** Let $\sigma \in S_k$ be a permutation with a cycle $c$. Then the cycle type of $(\prod_{i \in c} \rho_i(\sigma))$ is the same as that of $\sigma^{|c|}$. In other words, $\lambda\left(\prod_{i \in c} \rho_i(\sigma)\right) = \lambda(\sigma^{|c|})$.

**Proof.** Let $c = (c_1, c_2, \ldots, c_{|c|})$. First, we calculate:

$$\prod_{i=1}^{|c|} \rho_{c_i}(\sigma) = \rho_{c_{|c|}}(\sigma) \circ \cdots \circ \rho_{c_2}(\sigma) \circ \rho_{c_1}(\sigma)$$

$$= \rho_{c_{|c|}}(\sigma) \circ \cdots \circ \rho_{c_2}(\sigma)(a \mapsto \sigma(c_1 + a) - \sigma(c_1))$$

$$= \rho_{c_{|c|}}(\sigma) \circ \cdots \circ \rho_{c_2}(\sigma)(a \mapsto \sigma(c_2 + \sigma(c_1 + a) - \sigma(c_1)) - \sigma(c_2))$$

$$= \rho_{c_{|c|}}(\sigma) \circ \cdots \circ \rho_{c_2}(\sigma)(a \mapsto \sigma^2(c_1 + a) - \sigma^2(c_1))$$

$$\vdots$$

$$= a \mapsto \sigma^{|c|}(c_1 + a) - \sigma^{|c|}(c_1)$$

$$= \rho_{c_1}(\sigma^{|c|}).$$

But $c_1$ is a fixed point of $\sigma^{|c|}$, so by the result of Lemma 5.7, this has the same cycle structure as $\sigma^{|c|}$. \hfill \Box

From this and the fact that the terms of $X$-degree 1 in all $Z_{\overline{a}_k}^{X^Y}$ and $Z_{\overline{a}_{k+1}}^{X^Y}$ are equal (to $p_1[x]p_1[y]^{k+1}$), we can conclude that:

**Theorem 5.9.** $Z_{\overline{a}_k}^{X^Y}(\sigma)$ and $Z_{\overline{a}_{k+1}}^{X^Y}(\sigma)$ are class functions of $\sigma$ (that is, they are constant over permutations of fixed cycle type).

This will simplify computational enumeration of $k$-trees significantly, since the number of partitions of $k$ grows exponentially while the number of permutations of $[k]$ grows factorially.

### 5.2 $k$-trees: $a_k$

We now have all the pieces in hand to apply Theorem 5.1 to compute the cycle index of the species $a_k$ of general $k$-trees. Theorem 5.1 characterizes the cycle index of the generic $k$-tree species $a_k$ in terms of the cycle indices of the rooted species $a_k^X$, $a_k^Y$, and $a_k^{XY}$; Theorem 4.4 gives the cycle indices of these three rooted species in terms of the $\Gamma$-cycle indices $Z_{\overline{a}_k}^{X^Y}$, $Z_{\overline{a}_k}^{Y^X}$, and $Z_{\overline{a}_{k+1}}^{XY}$; and, finally, Theorems 5.5 and 5.6 give these $\Gamma$-cycle indices explicitly. By tracing the formulas in Eqs. (10) and (14) back through this sequence of functional relationships, we can conclude:

**Theorem 5.10** (Cycle index for the species of $k$-trees). For $a_k$ the species of general $k$-trees, $Z_{CT_k^{a_k}}$ as in Eq. (10), and $Z_{CT_k^{a_k+1}}$ as in Eq. (14) we have:

$$Z_{a_k} = \frac{1}{(k + 1)!} \sum_{\sigma \in S_{k+1}} Z_{CT_k^{a_k+1}}(\sigma) + \frac{1}{k!} \sum_{\sigma \in S_k} Z_{CT_k^{a_k}}(\sigma) - \frac{1}{k!} \sum_{\sigma \in S_k} Z_{CT_k^{a_k+1}}(\sigma)$$

Equation (15) in fact represents a recursive system of functional equations, since the formulas for the $\Gamma$-cycle indices of $CT_k^{a_k}$ and $CT_k^{a_k+1}$ are recursive. Computational methods can yield explicit enumerative results. However, a bit of care will allow us to reduce the computational complexity of this problem significantly.
6 Unlabeled enumeration and the generating function $\tilde{a}_k(x)$

Equation (15) in Theorem 5.10 gives a recursive formula for the cycle index of the (actionless) species $a_k$ of $k$-trees. The number of unlabeled $k$-trees with $n$ hedra is historically an open problem, but it is straightforward to extract their ordinary generating function from the cycle index $Z_{a_k}$ once it is computed. Actually computing terms of the cycle index in order to derive the coefficients of the generating function is, however, a computationally expensive process, since the cycle index is by construction a power series in two infinite sets of variables. The computational process can be simplified significantly by taking advantage of the relatively straightforward combinatorial structure of the structural decomposition used to derive the recursive formulas for the cycle index.

For a $\Gamma$-species $F$, the ordinary generating function $\tilde{F}_\gamma(x)$ counting unlabeled $\gamma$-invariant $F$-structures is given by

$$\tilde{F}(\gamma)(x) = Z_F(\gamma)(x, x^2, x^3, \ldots)$$

and the ordinary generating function for counting unlabeled $\ell/\tau$-structures is given by

$$\tilde{F}(x) = \frac{1}{|\Gamma|} \sum_{\gamma \in \Gamma} \tilde{F}(\gamma)(x).$$

These formula admits an obvious multisort extension, but we in fact wish to count $k$-trees with respect to just one sort of label (the $X$-labels on hedra), so we will not deal with multisort issues here. Each of our two-sort cycle indices can be converted to one-sort by substituting $a_i = \tilde{F}(i)(x)$ into the given formula. Thus, we have:

**Theorem 6.1.** For $\mathcal{C}_k^Y$ the $\mathcal{S}_k$-species of $Y$-rooted $k$-coding trees and $\mathcal{C}_k^{XY}$ the $\mathcal{S}_{k+1}$-species of edge-rooted $k$-coding trees, the corresponding $\Gamma$-ordinary generating functions are given by

$$\tilde{\mathcal{C}}_k^Y(\sigma)(x) = \exp \left( \sum_{n \geq 1} \frac{x^n}{n} \prod_{c \in C(\sigma^n)} \tilde{\mathcal{C}}_k^Y \left( \prod_{i \in c} \rho_i(\sigma^n) \right) (x^{|c|}) \right)$$

(16a)

and

$$\tilde{\mathcal{C}}_k^{XY}(\sigma)(x) = x \cdot \prod_{c \in C(\sigma)} \tilde{\mathcal{C}}_k^Y \left( \prod_{i \in c} \rho_i(\sigma) \right) (x^{|c|}).$$

(16b)

where $\tilde{\mathcal{C}}_k^Y$ is an $\mathcal{S}_k$-generating function and $\tilde{\mathcal{C}}_k^{XY}$ is an $\mathcal{S}_{k+1}$-generating function.

However, as a consequence of Theorem 5.9, we can simplify these expressions significantly. Let $\lambda$ be the function mapping each permutation to its cycle type interpreted as an integer partition. Then:

**Corollary 6.2.** For $\mathcal{C}_k^Y$ the $\mathcal{S}_k$-species of $Y$-rooted $k$-coding trees and $\mathcal{C}_k^{XY}$ the $\mathcal{S}_{k+1}$-species of edge-rooted $k$-coding trees, the corresponding $\Gamma$-ordinary generating functions are given by

$$\tilde{\mathcal{C}}_k^Y(\lambda)(x) = \exp \left( \sum_{n \geq 1} \frac{x^n}{n} \prod_{i \in \lambda^n} \tilde{\mathcal{C}}_k^Y(\lambda^{ni})(x^i) \right)$$

(17a)

and

$$\tilde{\mathcal{C}}_k^{XY}(\lambda)(x) = x \cdot \prod_{i \in \lambda} \tilde{\mathcal{C}}_k^Y(\lambda^i - \{1\})(x^i)$$

(17b)

where $\sum_{i \in \lambda}$ denotes a sum (and $\prod_{i \in \lambda}$ a product) over the parts $i$ of $\lambda$ taken with multiplicity, where $\lambda^i$ denotes the $i$th ‘partition power’ of $\lambda$ — that is, if $\sigma$ is any permutation of cycle type $\lambda$, then $\lambda^i$ denotes the cycle type of $\sigma^i$ — and where $f(\lambda)(x)$ denotes the value of $f(\sigma)(x)$ for any $\sigma$ of cycle type $\lambda$. 

14
As in Theorem 5.5, we have recursively-defined functional equations, but these are recursions of power series in a single variable, so computing their terms is much less computationally expensive. Also, as an immediate consequence of Theorem 5.9, we have that \( \overline{\mathcal{J}}^Y_k \) and \( \overline{\mathcal{J}}^{XY}_k \) are class functions of \( \sigma \), so we can restrict our computational attention to cycle-distinct permutations.

Moreover, the cycle index of the species \( a_k \), as seen in Eq. (15), is given simply in terms of quotients of the cycle indices of the two \( \Gamma \)-species \( \overline{\mathcal{J}}^Y_k \) and \( \overline{\mathcal{J}}^{XY}_k \). Thus, we also have:

**Theorem 6.3.** For \( a_k \) the species of \( k \)-trees and \( \overline{\mathcal{J}}^Y_k \) and \( \overline{\mathcal{J}}^{XY}_k \) as in Theorem 6.1, we have

\[
\tilde{a}_k(x) = \frac{1}{(k+1)!} \cdot \sum_{\sigma \in A_{k+1}} \overline{\mathcal{J}}^{XY}_k(\sigma)(x) + \frac{1}{k!} \cdot \sum_{\sigma \in A_k} \overline{\mathcal{J}}^Y_k(\sigma)(x) - \frac{1}{k!} \cdot \sum_{\sigma \in A_k} \overline{\mathcal{J}}^{XY}_k(\sigma)(x). \tag{18}
\]

Again, as a consequence of Theorem 5.9 by way of Corollary 6.2, we can instead write

**Corollary 6.4** (Generating function for unlabeled \( k \)-trees). For \( a_k \) the species of \( k \)-trees and \( \overline{\mathcal{J}}^Y_k \) and \( \overline{\mathcal{J}}^{XY}_k \) as in Corollary 6.2, we have

\[
\tilde{a}_k(x) = \sum_{\lambda \vdash k+1} \frac{1}{z_\lambda} \overline{\mathcal{J}}^{XY}_k(\lambda)(x) + \sum_{\lambda \vdash k} \frac{1}{z_\lambda} \overline{\mathcal{J}}^Y_k(\lambda)(x) \tag{19}
\]

This direct characterization of the ordinary generating function of unlabeled \( k \)-trees, while still recursive, is much simpler computationally than the characterization of the full cycle index in Eq. (15). For computation of the number of unlabeled \( k \)-trees, it is therefore much preferred. Classical methods for working with recursively-defined power series suffice to extract the coefficients quickly and efficiently. The results of some such explicit calculations are presented in Appendix A.

### 7 Special-case behavior for small \( k \)

Many of the complexities of the preceding analysis apply only for \( k \geq 3 \). In the cases \( k = 1 \) and \( k = 2 \), our analysis simplifies dramatically, and effectively reduces to previous work.

#### 7.1 Ordinary trees \( (k = 1) \)

When \( k = 1 \), an \( a_k \)-structure is merely an ordinary tree with \( X \)-labels on its edges and \( Y \)-labels on its vertices. There is no internal symmetry of the form that the actions of \( \mathcal{S}_k \) are intended to break. The actions of \( \mathcal{G}_2 \) act on ordinary trees rooted at a directed edge, with the nontrivial element \( \tau \in \mathcal{G}_2 \) acting by reversing this orientation. The resulting decomposition from the dissymmetry theorem in Theorem 2.1 and the recursive functional equations of Theorem 3.5 then clearly reduce to the classical dissymmetry analysis of ordinary trees.

#### 7.2 2-trees

When \( k = 2 \), there is a nontrivial symmetry at fronts (edges); two triangles may be joined at an edge in two distinct ways. The imposition of a coherent orientation on a 2-tree by directing one of its edges breaks this symmetry; the action of \( \mathcal{G}_2 \) by reversal of these orientations gives unoriented 2-trees as its orbits. The defined action of \( \mathcal{G}_3 \) on edge-rooted oriented triangles is simply the classical action of the dihedral group \( D_6 \) on a triangle, and its orbits are unoriented, unrooted triangles. We further note that \( \rho_t \) is the trivial map on \( \mathcal{G}_2 \) and that \( \rho_{t}\sigma = (12) \) for \( \sigma \in \mathcal{G}_3 \) if and only if \( \sigma \) is an odd permutation, both regardless of \( i \). We then have that:

\[
Z^{\mathcal{G}_2}_2 = p_1[y] \cdot Z_C \circ \left(p_1[x] \cdot \prod_{c \in C} Z^{\mathcal{G}_2}_2(c)(p_{2|c}[x], p_{2|c}[y], \ldots, p_{2|c}[y], p_{2|c}[y], \ldots) \right) \tag{20a}
\]
and
\[ Z^{S_3}_{ct} = p_1[x] \cdot \prod_{c \in C(\sigma)} Z^{S_2}_{ct} (\rho(\sigma)^{|c|}) (p_{|c|}[x], p_{2|c|}[x], \ldots; p_{|c|}[y], p_{2|c|}[y], \ldots). \] (20b)

where, by abuse of notation, we let \( \rho \) represent any \( \rho_i \). By the previous, the argument \( \rho(\sigma)^{|c|} \) in Eq. (20b) is \( \tau \) if and only if \( \sigma \) is an odd permutation and \( c \) is of odd length. This analysis and the resulting formulas for the cycle index \( Z_{a_2} \) are essentially equivalent to those derived in [4].

A Enumerative tables

With the recursive functional equations for cycle indices of Section 5, we can calculate the explicit cycle index for the species \( a_k \) to any finite degree we choose using computational methods; this cycle index can then be used to enumerate both unlabeled and labeled (at fronts, hedra, or both) \( k \)-trees up to a specified number \( n \) of hedra (or, equivalently, \( kn + 1 \) fronts). We have done so here for \( k \leq 7 \) and \( n \leq 30 \) using Sage 5.0 [13] using code available in Appendix B. The resulting values appear in Table 1.

We note that both unlabeled and hedron-labeled enumerations of \( k \)-trees stabilize in \( k \):

**Theorem A.1.** For \( k \geq n - 2 \), the numbers of unlabeled and hedron-labeled \( k \)-trees with \( n \) hedra are independent of \( k \).

**Proof.** We show that the species \( a_k \) and \( a_{k+1} \) have contact up to order \( k + 2 \) by explicitly constructing a natural bijection. We note that in a \((k + 1)\)-tree with no more than \( k + 2 \) hedra, there will exist at least one vertex which is common to all hedra. For any \( k \)-tree with no more than \( k + 2 \) hedra, we can construct a \((k + 1)\)-tree with the same number of hedra by adding a single vertex and connecting it by edges to every existing vertex; we can then pass labels up from the \((k + 1)\)-cliques which are the hedra of the \( k \)-tree to the \((k + 2)\)-cliques which now sit over them. The resulting graph will be a \((k + 1)\)-tree whose \((k + 1)\)-tree hedra are adjacent exactly when the \( k \)-tree hedra they came from were adjacent. Therefore, any two distinct \( k \)-trees will pass to distinct \((k + 1)\)-trees. Similarly, for any \((k + 1)\)-tree with no more than \( k + 2 \) hedra, choose one of the vertices common to all the hedra and remove it, passing the labels of \((k + 1)\)-tree hedra down to the \( k \)-tree hedra constructed from them; again, adjacency of hedra is preserved. This of course creates a \( k \)-tree, and for distinct \((k + 1)\)-trees the resulting \( k \)-trees will be distinct. Moreover, by symmetry the result is independent of the choice of common vertex, in the case there is more than one. □

However, thus far we have neither determined a direct method for computing these stabilization numbers nor identified a straightforward combinatorial characterization of the structures they represent.
Table 1: Enumerative data for $k$-trees with $n$ hedra

(a) $k = 1$

| $n$ | Unlabeled 1-trees |
|-----|-------------------|
| 0   | 1                 |
| 1   | 1                 |
| 2   | 1                 |
| 3   | 2                 |
| 4   | 3                 |
| 5   | 6                 |
| 6   | 11                |
| 7   | 23                |
| 8   | 47                |
| 9   | 106               |
| 10  | 235               |
| 11  | 551               |
| 12  | 1301              |
| 13  | 3159              |
| 14  | 7741              |
| 15  | 19320             |
| 16  | 48629             |
| 17  | 123867            |
| 18  | 317955            |
| 19  | 823065            |
| 20  | 2144505           |
| 21  | 5623756           |
| 22  | 14828074          |
| 23  | 39299897          |
| 24  | 104636890         |
| 25  | 279793450         |
| 26  | 751065460         |
| 27  | 2023443032        |
| 28  | 5469566585        |
| 29  | 14830871802       |
| 30  | 40330829030       |

(b) $k = 2$

| $n$ | Unlabeled 2-trees |
|-----|-------------------|
| 0   | 1                 |
| 1   | 1                 |
| 2   | 2                 |
| 3   | 3                 |
| 4   | 5                 |
| 5   | 12                |
| 6   | 39                |
| 7   | 136               |
| 8   | 529               |
| 9   | 2171              |
| 10  | 9368              |
| 11  | 41534             |
| 12  | 188942            |
| 13  | 874906            |
| 14  | 4115060           |
| 15  | 19602156          |
| 16  | 94419351          |
| 17  | 459183768         |
| 18  | 2252217207        |
| 19  | 11130545494       |
| 20  | 55382155396       |
| 21  | 277255622646      |
| 22  | 1395731021610     |
| 23  | 7061871805974     |
| 24  | 35896206800034     |
| 25  | 183241761631584   |
| 26  | 939081790240231   |
| 27  | 4830116366008952  |
| 28  | 2492717592036185  |
| 29  | 129047003236769110|
| 30  | 670024248072778235|
Enumerative data for $k$-trees with $n$ hedra, continued

(c) $k = 3$

| $n$ | Unlabeled 3-trees |
|-----|-------------------|
| 0   | 1                 |
| 1   | 1                 |
| 2   | 1                 |
| 3   | 2                 |
| 4   | 5                 |
| 5   | 15                |
| 6   | 58                |
| 7   | 275               |
| 8   | 1505              |
| 9   | 9003              |
| 10  | 56931             |
| 11  | 372973            |
| 12  | 2506312           |
| 13  | 17165954          |
| 14  | 119398333         |
| 15  | 841244274         |
| 16  | 5993093551        |
| 17  | 43109340222       |
| 18  | 312747109787      |
| 19  | 2286190318744     |
| 20  | 16826338257708    |
| 21  | 124605344758149   |
| 22  | 92791020739261    |
| 23  | 6945172081954449  |
| 24  | 5222528386702922  |
| 25  | 394398440097305861|
| 26  | 2990207055800156659|
| 27  | 22753619938517594709|
| 28  | 173727411594289881739|
| 29  | 1330614569159767263501|
| 30  | 10221394007530945428347|

(d) $k = 4$

| $n$ | Unlabeled 4-trees |
|-----|-------------------|
| 0   | 1                 |
| 1   | 1                 |
| 2   | 2                 |
| 3   | 2                 |
| 4   | 5                 |
| 5   | 15                |
| 6   | 64                |
| 7   | 331               |
| 8   | 2150              |
| 9   | 15817             |
| 10  | 127194            |
| 11  | 107763            |
| 12  | 9466983           |
| 13  | 85252938          |
| 14  | 782238933         |
| 15  | 7283470324        |
| 16  | 68639621442       |
| 17  | 653492361220      |
| 18  | 6276834750665     |
| 19  | 6075938837299     |
| 20  | 592227182125701   |
| 21  | 580846697002391   |
| 22  | 57289008242377068 |
| 23  | 567939935463185078|
| 24  | 565670014851208902|
| 25  | 56583199285317631541|
| 26  | 568236762643725657852|
| 27  | 5727423267612393252616|
| 28  | 57924486783495226147615|
| 29  | 58767209044784037304025|
| 30  | 5979782184127687211698807|
Enumerative data for \(k\)-trees with \(n\) hedra, continued

\[(e)\] \(k = 5\)

\[
\begin{array}{|c|c|}
\hline
n & \text{Unlabeled 5-trees} \\
\hline
0 & 1 \\
1 & 1 \\
2 & 2 \\
3 & 4 \\
4 & 8 \\
5 & 16 \\
6 & 32 \\
7 & 64 \\
8 & 128 \\
9 & 256 \\
10 & 512 \\
11 & 1024 \\
12 & 2048 \\
13 & 4096 \\
14 & 8192 \\
15 & 16384 \\
16 & 32768 \\
17 & 65536 \\
18 & 131072 \\
19 & 262144 \\
20 & 524288 \\
21 & 1048576 \\
22 & 2097152 \\
23 & 4194304 \\
24 & 8388608 \\
25 & 16777216 \\
26 & 33554432 \\
27 & 67108864 \\
28 & 134217728 \\
29 & 268435456 \\
30 & 536870912 \\
\hline
\end{array}
\]

\[(f)\] \(k = 6\)

\[
\begin{array}{|c|c|}
\hline
n & \text{Unlabeled 6-trees} \\
\hline
0 & 0 \\
1 & 1 \\
2 & 2 \\
3 & 6 \\
4 & 20 \\
5 & 70 \\
6 & 252 \\
7 & 924 \\
8 & 3432 \\
9 & 12870 \\
10 & 48620 \\
11 & 184756 \\
12 & 705432 \\
13 & 2764156 \\
14 & 1046080 \\
15 & 3876000 \\
16 & 14307840 \\
17 & 52003272 \\
18 & 195047560 \\
19 & 715017760 \\
20 & 2635013280 \\
21 & 9694845760 \\
22 & 35349829120 \\
23 & 128990792000 \\
24 & 479001625760 \\
25 & 1711878375936 \\
26 & 6284513057360 \\
27 & 22828107200000 \\
28 & 83933220800000 \\
29 & 308888883200000 \\
30 & 1124000728000000 \\
\hline
\end{array}
\]
Enumerative data for \(k\)-trees with \(n\) hedra, continued

\[
(g) \ k = 7
\]

| \(n\) | Unlabeled 7-trees |
|-------|-------------------|
| 0     | 1                 |
| 1     | 1                 |
| 2     | 1                 |
| 3     | 2                 |
| 4     | 5                 |
| 5     | 15                |
| 6     | 64                |
| 7     | 342               |
| 8     | 2344              |
| 9     | 19137             |
| 10    | 181098            |
| 11    | 192215            |
| 12    | 22472875          |
| 13    | 284556458         |
| 14    | 3849827955       |
| 15    | 5497480527        |
| 16    | 819865209740      |
| 17    | 12655913153775    |
| 18    | 200748351368185   |
| 19    | 3253193955012557  |
| 20    | 53619437319817482 |
| 21    | 895778170144927928 |
| 22    | 15129118461773051724 |
| 23    | 257812223121779545108 |
| 24    | 4426056869082751747930 |
| 25    | 76463433541541506345648 |
| 26    | 1328088941166844504424628 |
| 27    | 23175796698013212039339479 |
| 28    | 406103563562864890670029228 |
| 29    | 7142350290468621849814034057 |
| 30    | 126034923903699365819345698783 |
Enumerative data for $k$-trees with $n$ hedra, continued

(h) $k = 8$

| $n$ | Unlabeled 8-trees |
|-----|--------------------|
| 0   | 1                  |
| 1   | 1                  |
| 2   | 1                  |
| 3   | 2                  |
| 4   | 5                  |
| 5   | 15                 |
| 6   | 64                 |
| 7   | 342                |
| 8   | 2344               |
| 9   | 19137              |
| 10  | 181204             |
| 11  | 1926782            |
| 12  | 22638677           |
| 13  | 289742922          |
| 14  | 3996857019         |
| 15  | 58854922207        |
| 16  | 916955507587       |
| 17  | 14988769972628     |
| 18  | 255067524402905    |
| 19  | 4487202163529135   |
| 20  | 81112295567987808  |
| 21  | 149887411789828574 |
| 22  | 2819596539534035096|
| 23  | 538126404726276758908|
| 24  | 10391826059632904271057|
| 25  | 202624626664206041379718|
| 26  | 3982593421723767068438772|
| 27  | 78804180647706388187446055|
| 28  | 1568191570016583843925943321|
| 29  | 31359266621157738864915907470|
| 30  | 629755261439815181073415721542|
Enumerative data for $k$-trees with $n$ hedra, continued

(i) $k = 9$

| $n$ | Unlabeled 9-trees |
|-----|-------------------|
| 0   | 1                 |
| 1   | 1                 |
| 2   | 1                 |
| 3   | 2                 |
| 4   | 5                 |
| 5   | 15                |
| 6   | 64                |
| 7   | 342               |
| 8   | 2344              |
| 9   | 19137             |
| 10  | 181204            |
| 11  | 1927017           |
| 12  | 22652254          |
| 13  | 290351000         |
| 14  | 401997352         |
| 15  | 59642496465       |
| 16  | 941751344429      |
| 17  | 15724551551655    |
| 18  | 275926445572426   |
| 19  | 5057692869843759  |
| 20  | 96275031338911591 |
| 21  | 189267812366295682 |
| 22  | 38234411627616084843 |
| 23  | 790120238796588845615 |
| 24  | 16638524087850961727575 |
| 25  | 355878246778832856290372 |
| 26  | 7710423952280397990026132 |
| 27  | 168843592748278228259801752 |
| 28  | 373028552085543827693340329 |
| 29  | 83027821492843727307516904184 |
| 30  | 1859625249087075723295908757282 |
Enumerative data for $k$-trees with $n$ hedra, continued

(j) $k = 10$

| $n$ | Unlabeled 10-trees |
|-----|--------------------|
| 0   | 1                  |
| 1   | 1                  |
| 2   | 1                  |
| 3   | 2                  |
| 4   | 5                  |
| 5   | 15                 |
| 6   | 64                 |
| 7   | 342                |
| 8   | 2344               |
| 9   | 19137              |
| 10  | 181204             |
| 11  | 1927017            |
| 12  | 22652805           |
| 13  | 290391147          |
| 14  | 4022154893         |
| 15  | 59741455314        |
| 16  | 945737514583       |
| 17  | 15871943695637     |
| 18  | 281035862707569    |
| 19  | 5226147900656616   |
| 20  | 101612006684523937 |
| 21  | 205642512391014429 |
| 22  | 43127730369661586804 |
| 23  | 933229734601780336024 |
| 24  | 20749443766669472108394 |
| 25  | 472211306357077710523863 |
| 26  | 10961384502758318928846970 |
| 27  | 258737420965101611169934566 |
| 28  | 6193917232793763076872721853 |
| 29  | 150039339181032274342778699887 |
| 30  | 3670778410024403632885217999313 |
B Code listing

The recursive functional equations in Eqs. (17a), (17b) and (19) characterize the ordinary generating function $\tilde{a}_k(x)$ for unlabeled general $k$-trees. Code to compute the coefficients of this generating function using the computer algebra system Sage 5.0 [13] explicitly follows in listing 1. Specifically, the generating function for unlabeled $k$-trees may be computed to degree $n$ by copying the included code into a Sage notebook, modifying the final line with the desired values of $k$ and $n$, and executing.

This code uses the class-function optimization of Theorem 5.9 extensively; as a result, it is able to compute the number of $k$-trees on up to $n$ hedra quickly even for relatively large $k$ and $n$. For example, the first thirty terms of the generating function for 8-trees in Table 1b were computed on a modern desktop-class computer in approximately two minutes.

Listing 1: Sage code to compute numbers of $k$-trees

```python
# Set up a ring of formal power series
psr = PowerSeriesRing(QQ, 'x')
x = psr.gen()

# Compute the generating function for unlabeled Y-rooted k-trees fixed by permutations of a given cycle type mu.
# Note that mu should partition k
@cached_function
def unlY(mu, n):
    if n <= 0:
        return psr(1)
    else:
        ystretch = lambda c, part: unlY(Partition(partition_power(part, c)), floor((n-1)/c)).subs({x:x**c})
        descendant_pseries = lambda part: prod ystretch(c, part) for c in part
        return sum(x**i/i * descendant_pseries(partition_power(mu, i)).subs({x:x**i}) for i in xrange(1, n+1)).exp(n+1)

# Compute the generating function for unlabeled XY-rooted k-trees fixed by permutations of a given cycle type mu.
# Note that mu should partition k+1
@cached_function
def unlXY(mu, n):
    if n <= 0:
        return psr(0)
    else:
        ystretch = lambda c: unlY(Partition(partition_power(mu, c)[:-1]), floor((n-1)/c)).subs({x:x**c})
        return (x * prod ystretch(c) for c in mu).add_bigoh(n+1)

# Compute the generating functions for unlabeled X-, Y-, and XY-rooted k-trees using quotients
ax = lambda k, n: sum(1/mu.aut() * unlXY(mu, n) for mu in Partitions(k+1))
ay = lambda k, n: sum(1/mu.aut() * unlY(mu, n) for mu in Partitions(k))
axy = lambda k, n: sum(1/mu.aut() * unlXY(Partition(mu + [1]), n) for mu in Partitions(k))

# Compute the generating function for unlabeled un-rooted k-trees using the dissymmetry theorem
a = lambda k, n: ax(k, n) + ay(k, n) - axy(k, n)

# Print the result
# ALERT: User must substitute values for k and n (number of hedra)
print a(kval, nval)
```

Acknowledgments

The author wishes to express his profound gratitude to Ira Gessel, who served as advisor and committee chair for the dissertation in which this research originated. His guidance, advice, and insightful mentorship were invaluable throughout that process.
The author also offers his thanks to an anonymous referee, whose close reading and thorough commentary led to the correction of numerous small errors and oversights and whose broader recommendations led to significant improvement of the structure and narrative flow of the paper as a whole.
References

[1] Lowell W. Beineke and Raymond E. Pippert. Multidimensional bipartite trees. *Discrete Mathematics*, 272:17–26, 2003.

[2] L.W. Beineke and R.E. Pippert. The number of labeled k-dimensional trees. *Journal of Combinatorial Theory*, 6(2):200–205, 1969.

[3] F. Bergeron, G. Labelle, and P. Leroux. *Combinatorial species and tree-like structures*, volume 67 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, Cambridge, 1998. Translated from the 1994 French original by Margaret Readdy, With a foreword by Gian-Carlo Rota.

[4] Tom Fowler, Ira Gessel, Gilbert Labelle, and Pierre Leroux. The specification of 2-trees. *Adv. in Appl. Math.*, 28(2):145–168, 2002.

[5] Andrew Gainer-Dewar. *Γ-species, quotients, and graph enumeration*. PhD thesis, Brandeis University, 2012.

[6] F. Harary and E. Palmer. *Graphical Enumeration*. Academic Press, New York, 1973.

[7] F. Harary and E.M. Palmer. On acyclic simplicial complexes. *Mathematika*, 15:115–122, 1968.

[8] Anthony Henderson. Species over a finite field. *J. Algebraic Combin.*, 21(2):147–161, 2005.

[9] Johannes Köbler and Sebastian Kuhnert. The isomorphism problem for k-trees is complete for logspace. In *Proceedings of the 34th International Symposium on Mathematical Foundations of Computer Science*, pages 537–548, Berlin, Heidelberg, 2009. Springer-Verlag.

[10] J.W. Moon. The number of labeled k-trees. *Journal of Combinatorial Theory*, 6(2):196–199, 1969.

[11] E. Palmer. On the number of labeled 2-trees. *Journal of Combinatorial Theory*, 6:206–207, 1969.

[12] E. Palmer and R. Read. On the number of plane 2-trees. *Journal of the London Mathematical Society*, 6:583–592, 1973.

[13] W.A. Stein et al. *Sage Mathematics Software (Version 5.0)*. The Sage Development Team, 2011. [http://www.sagemath.org](http://www.sagemath.org).