Γ-species, Quotients, and Graph Enumeration

A Dissertation

Presented to
The Faculty of the Graduate School of Arts and Sciences
Brandeis University
Department of Mathematics
Ira Gessel, Advisor

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

by
Andrew Gainer

May, 2012

Dedication

For my friends, who keep me sane.

Acknowledgments

Without Ira's brilliant and understanding mentorship, this research would not have been possible.

Without Susan's friendship and guidance or the camaraderie of the members of my cohort, I never would have made it to the point of doing graduate research.

Without the encouragement David, Margaret, and Kim gave me, I might never have discovered mathematical research at all.

Without my parents' patient and generous support or the forbearance of my many teachers from childhood into my graduate years, I would never even have discovered the academy.

Without Janet's affection, fellowship, and tolerance of my eccentricities, my progress in these last years—and my spirits—would have been greatly diminished.

My life and this work are gift from and a testament to everyone whom I have known. Thank you all.

Abstract

Γ-species, Quotients, and Graph Enumeration

A dissertation presented to the Faculty of the
Graduate School of Arts and Sciences of Brandeis
University, Waltham, Massachusetts

by Andrew Gainer

The theory of Γ-species is developed to allow species-theoretic study of quotient structures in a categorically rigorous fashion. This new approach is then applied to two graph-enumeration problems which were previously unsolved in the unlabeled case—bipartite blocks and general k-trees.

Preface

Historically, the algebra of generating functions has been a valuable tool in enumerative combinatorics. The theory of combinatorial species uses category theory to justify and systematize this practice, making clear the connections between structural manipulations of some objects of interest and algebraic manipulations of their associated generating functions. The notion of 'quotient' enumeration (that is, of counting orbits under some group action) has been applied in species-theoretic contexts, but methods for doing so have largely been ad-hoc. We will contribute a species-compatible way for keeping track of the way a group Γ acts on structures of a species F, yielding what we term a Γ-species, which has the sort of synergy of algebraic and structural data that we expect from species. We will then show that it is possible to extract information about the Γ-orbits of such a Γ-species and harness this new method to attack several unsolved problems in graph enumeration—in particular, the isomorphism classes of nonseparable bipartite graphs and k-trees (that is, 'unlabeled' bipartite blocks and k-trees).

It is assumed that the reader of this thesis is familiar with the classical theory of groups and that he has encountered at least the basic vocabularies of category theory and graph theory. Results in these fields which are not original to this thesis will either be referenced from the literature or simply assumed, depending on the degree to which they are part of the standard body of knowledge one acquires when studying those disciplines.

In the first chapter, we outline the theory of species, develop several classical methods, and introduce the notion of a Γ-species. In the second chapter, we apply these techniques to the enumeration of unlabeled vertex-2-connected bipartite graphs, a historically open problem. In the third chapter, we apply these techniques to the more complex problem of the enumeration of unlabeled general k-trees, also historically unsolved. Finally, in an appendix we discuss algebraic and computational methods which allow species-theoretical insights to be translated into explicit algorithmic techniques for enumeration.

Contents

List of Figures

List of Tables

CHAPTER 1

The theory of species

1.1. Introduction

Many of the most important historical problems in enumerative combinatorics have concerned the difficulty of passing from 'labeled' to 'unlabeled' structures. In many cases, the algebra of generating functions has proved a powerful tool in analyzing such problems. However, the general theory of the association between natural operations on classes of such structures and the algebra of their generating functions has been largely ad-hoc. André Joyal's introduction of the theory of combinatorial species in **[15]** provided the groundwork to formalize and understand this connection. A full, pedagogical exposition of the theory of species is available in **[3]**, so we here present only an outline, largely tracking that text.

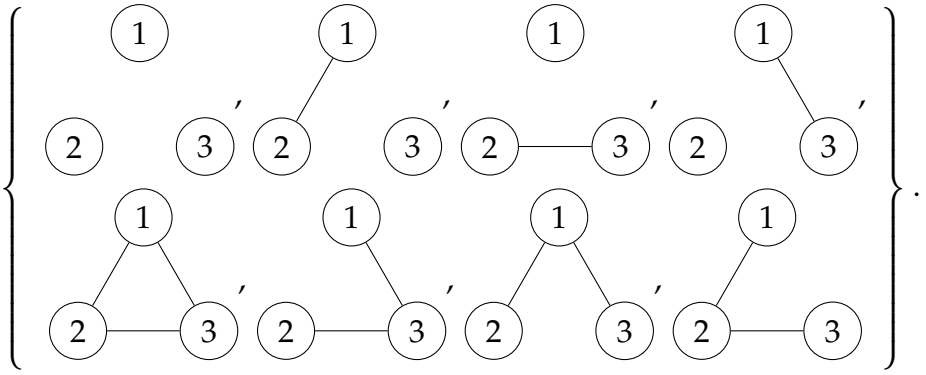
To begin, we wish to formalize the notion of a 'construction' of a structure of some given class from a set of 'labels', such as the construction of a graph from its vertex set or or that of a linear order from its elements. The language of category theory will allow us capture this behavior succinctly yet with full generality:

Definition 1.1.1. Let **FinBij** be the category of finite sets with bijections and **FinSet** be the category of finite sets with set maps. Then a *species* is a functor $F : \mathbf{FinBij} \to \mathbf{FinSet}$. For a set A and a species F, an element of $F[A]$ is an *F-structure on A*. Moreover, for a species F and a bijection $\phi : A \to B$, the bijection $F[\phi] : F[A] \to F[B]$ is the *F-transport of ϕ*.

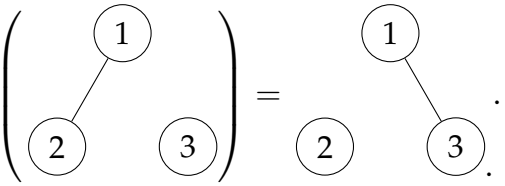
A species functor F simply associates to each set A another set $F[A]$ of its F-structures; for example, for $\mathcal{S}$ the species of permutations, we associate to some set A the set $\mathcal{S}[A] = \mathrm{Bij}(A)$ of self-bijections (that is, permutations as maps) of A. This association of label set A to the set $F[A]$ of all F-structures over A is fundamental throughout combinatorics, and functorality is simply the requirement that we may carry maps on the label set through the construction.

Example 1.1.2. Let $\mathcal{G}$ denote the species of simple graphs labeled at vertices. Then, for any finite set A of labels, $G[A]$ is the set of simple graphs with $|A|$ vertices labeled by the elements of A. For example, for label set $A = [3] = \{1, 2, 3\}$, there are eight graphs in $\mathcal{G}[A]$, since there are $\binom{3}{2} = 3$ possible edges and thus $2^3 = 8$

ways to choose a subset of those edges:

$\mathcal{G}[\{1,2,3\}] =$ 

The symmetric group $\mathfrak{S}_3$ acts on the set $[3]$ as permutations. Consider the permutation (23) that interchanges 2 and 3 in $[3]$. Then $\mathcal{G}[(23)]$ is a permutation on the set $\mathcal{G}[\{1,2,3\}]$; for example,

$\mathcal{G}[(23)]$ 

Since the image of a bijection under such a functor is necessarily itself a bijection, many authors instead simply define a species as a functor $F : \mathbf{FinBij} \to \mathbf{FinBij}$. Our motivation for using this definition instead will become clear in Section 1.5.

Note that, having defined the species F to be a functor, we have the following properties:

- for any two bijections $\alpha : A \to B$ and $\beta : B \to C$, we have $F[\alpha \circ \beta] = F[\alpha] \circ F[\beta]$, and
- for any set A, we have $F[\mathrm{Id}_A] = \mathrm{Id}_{F[A]}$.

Accordingly, we (generally) need not concern ourselves with the details of the set A of labels we consider, so we will often restrict our attention to a canonical label set $[n] := \{1, 2, \ldots, n\}$ for each cardinality n. Moreover, the permutation group $\mathfrak{S}_A$ on any given set A acts by self-bijections of A and induces *automorphisms* of F-structures for a given species F. The orbits of F-structures on A under the induced action of $\mathfrak{S}_A$ are then exactly the 'unlabeled' structures of the class F, such as unlabeled graphs.

Finally, we note that it is often natural to speak of maps between classes of combinatorial structures, and that these maps are sometimes combinatorially 'natural'. For example, we might wish to map the species of trees into the species of general graphs by embedding; to map the species of connected bicolored graphs to the species of connected bipartite graphs by forgetting some color information; or the species of graphs to the species of sets of connected graphs by identification.

These maps are all 'natural' in the sense that they are explicitly structural and do not reference labels; thus, at least at a conceptual level, they are compatible with the motivating ideas of species. We can formalize this notion in the language of categories:

Definition 1.1.3. Let F and G be species. A *species map* ϕ is a natural transformation $\phi : F \to G$ — that is, an association to each set $A \in \mathbf{FinBij}$ of a set map $\phi_A \in \mathbf{FinSet}$ such that the following diagram commutes:

$$\begin{array}{ccc} F[A] & \xrightarrow{F[\sigma]} & F[B] \\ \downarrow{\scriptstyle \phi_A} & & \downarrow{\scriptstyle \phi_B} \\ G[A] & \xrightarrow{G[\sigma]} & G[B] \end{array}$$

We call the set map ϕ_A the *A component of ϕ* or the *component of ϕ at A*.

Such species maps may capture the idea that two species are essentially 'the same' or that one 'contains' or 'sits inside' another.

Definition 1.1.4. Let F and G be species and $\phi : F \to G$ a species map between them. In the case that the components ϕ_A are all bijections, we say that ϕ is a *species isomorphism* and that F and G are *isomorphic*. In the case that the components ϕ_A are all injections, we say that ϕ is a *species embedding* and that *F embeds in G* (denoted $\phi : F \hookrightarrow G$). Likewise, in the case that the components ϕ_A are all surjections, we say that ϕ is a *species covering* and that *F covers G* (denoted $\phi : F \twoheadrightarrow G$).

With the full power of the language of categories, we may make the following more general observation:

Note 1.1.5. Let **Spc** denote the functor category of species; that is, define $\mathbf{Spc} := \mathbf{FinSet}^{\mathbf{FinBij}}$, the category of functors from **FinBij** to **FinSet**. Species maps as defined in Definition 1.1.3 are natural transformations of these functors and thus are exactly the morphisms of **Spc**.

It is a classical theorem of category theory (cf. [**17**]) that the epi- and monomorphisms of a functor category are exactly those whose components are epi- and monomorphisms in the target category if the target category has pullbacks and pushouts. Since **FinSet** is such a category, species embeddings and species coverings are precisely the epi- and monomorphisms of the functor category **Spc**. Species isomorphisms are of course the categorical isomorphisms in **Spc**.

In the case that F and G are isomorphic species, we will often simply write $F = G$, since they are combinatorially equivalent; some authors instead use $F \simeq G$, reserving the notation of equality for the much stricter case that additionally

requires that $F[A] = G[A]$ as sets for all A. The notions of species embedding and species covering are original to this work.

Example 1.1.6. In the motivating examples from above:

- The species $\mathfrak{a}$ of trees *embeds* in the species $\mathcal{G}$ of graphs by the map which identifies each tree with itself as a graph, since any two distinct trees are distinct as graphs.
- The species $\mathcal{BC}$ of bicolored graphs *covers* the species $\mathcal{BP}$ of bipartite graphs by the map which sends each bicolored graph to its underlying bipartite graph, since every bipartite graph has at least one bicoloring.
- The species $\mathcal{G}$ of graphs is *isomorphic* with the species $\mathcal{E}\left(\mathcal{G}^{\mathcal{C}}\right)$ of sets of connected graphs by the map which identifies each graph with its set of connected components, since this decomposition exists uniquely.

1.2. Cycle indices and species enumeration

In classical enumerative combinatorics, formal power series known as 'generating functions' are used extensively for keeping track of enumerative data. In this spirit, we now associate to each species a formal power series which counts structures with respect to their automorphisms, which will prove to be significantly more powerful:

Definition 1.2.1. For a species F, define its *cycle index series* to be the power series (1)

$$Z_F(p_1, p_2, \ldots) := \sum_{n \geqslant 0} \frac{1}{n!} \Big(\sum_{\sigma \in \mathfrak{S}_n} \operatorname{fix}(F[\sigma])\, p_1^{\sigma_1} p_2^{\sigma_2} \ldots \Big) = \sum_{n \geqslant 0} \frac{1}{n!} \Big(\sum_{\sigma \in \mathfrak{S}_n} \operatorname{fix}(F[\sigma])\, p_\sigma \Big)$$

where $\operatorname{fix}(F[\sigma]) := |\{s \in F[A] : F[\sigma](s) = s\}|$, where σ_i is the number of i-cycles of σ, and where p_i are indeterminates. (That is, $\operatorname{fix}(F[\sigma])$ is the *number* of F-structures fixed under the action of the transport of σ.) We will make extensive use of the compressed notation $p_\sigma = p_1^{\sigma_1} p_2^{\sigma_2} \ldots$ hereafter.

In fact, by functorality, $\operatorname{fix}(F[\sigma])$ is a class function[1] on permutations $\sigma \in \mathfrak{S}_n$. Accordingly, we can instead consider all permutations of a given cycle type at once. It is a classical theorem that conjugacy classes of permutations in $\mathfrak{S}_n$ are indexed by partitions $\lambda \vdash n$, which are defined as multisets of natural numbers whose sum is n. In particular, conjugacy classes are determined by their cycle type, the multiset of the lengths of the cycles, which may clearly be identified bijectively with partitions of n. For a given partition $\lambda \vdash n$, there are $n!/z_\lambda$ permutations of cycle type λ, where $z_\lambda := \prod_i i^{\lambda_i} \lambda_i!$ where λ_i denotes the multiplicity of i in λ..

[1] That is, the value of $\operatorname{fix}(F[\sigma])$ will be constant on conjugacy classes of permutations, which we note are exactly the sets of permutations of fixed cycle type.

Thus, we can instead write

$$Z_F(p_1, p_2, \ldots) := \sum_{n \geqslant 0} \sum_{\lambda \vdash n} \operatorname{fix}(F[\lambda]) \frac{p_1^{\lambda_1} p_2^{\lambda_2} \cdots}{z_\lambda} = \sum_{n \geqslant 0} \sum_{\lambda \vdash n} \operatorname{fix}(F[\lambda]) \frac{p_\lambda}{z_\lambda} \tag{2}$$

for $\operatorname{fix} F[\lambda] := \operatorname{fix} F[\sigma]$ for some choice of a permutation σ of cycle type λ. Again, we will make extensive use of the notation $p_\lambda = p_\sigma$ hereafter.

That the cycle index Z_F usefully characterizes the enumerative structure of the species F may not be clear. However, as the following theorems show, both labeled and unlabeled enumeration are immediately possible once the cycle index is in hand. Recall that, for a given sequence $a = (a_0, a_1, a_2, \ldots)$, the *ordinary generating function*[2] of a is the formal power series $\tilde{A}(x) = \sum_{i=0}^{\infty} a_i x^i$ and the *exponential generating function* is the formal power series $A(x) = \sum_{i=1}^{\infty} \frac{1}{i!} a_i x^i$. The scaling factor of $\frac{1}{n!}$ in the exponential generating function is convenient in many contexts; for example, it makes differentiation of the generating function a combinatorially-significant operation. The cycle index of a species is then directly related to two important generating functions:

THEOREM 1.2.2. *The exponential generating function $F(x)$ of labeled F-structures is given by*

$$F(x) = Z_f(x, 0, 0, \ldots). \tag{3}$$

THEOREM 1.2.3. *The ordinary generating function $\tilde{F}(x)$ of unlabeled F-structures is given by*

$$\tilde{F}(x) = Z_F(x, x^2, x^3, \ldots). \tag{4}$$

Proofs of both theorems are found in [**3**, §1.2]. In essence, Eq. (3) counts each labeled structure exactly once (as a fixed point of the trivial automorphism on $[n]$) with a factor of $1/n!$, while Eq. (4) simply counts orbits *à la* Burnside's Lemma. In cases where the unlabeled enumeration problem is interesting, it is generally more challenging than the labeled enumeration of the same structures, since the characterization of isomorphism in a species may be nontrivial to capture in a generating function. If, however, we can calculate the complete cycle index for a species, both labeled and unlabeled enumerations immediately follow.

The use of p_i for the variables instead of the more conventional x_i alludes to the theory of symmetric functions, in which p_i denotes the power-sum functions $p_i = \sum_j x_j^i$, which form an important basis for the ring Λ of symmetric functions. When the p_i are understood as symmetric functions rather than simply indeterminates, additional Pólya-theoretic enumerative information is exposed. In particular, the symmetric function in x-variables underlying a cycle index in p-variables

[2] Although these are called 'functions' for historical reasons, convergence of these formal power series is often not of immediate interest.

may be said to count *partially*-labeled structures of a given species, where the coefficient on a monomial $\prod x_i^{\alpha_i}$ counts structures with α_i labels of each sort i. This serves to explain why the coefficients of powers of $p_1 = \sum_i x_i$ counts labeled structures (where the labels must all be distinct) and why the automorphism types of structures are enumerated by $Z_f\left(x, x^2, x^3, \cdots\right)$, which allows clusters of labels to be the same. Another application of the theory of symmetric functions to the cycle indices of species may be found in **[7]**.

A more detailed exploration of the history of cycle index polynomials and their relationship to classical Pólya theory may be found in **[18]**.

Of course, it is not always obvious how to calculate the cycle index of a species directly. However, in cases where we can decompose a species as some combination of simpler ones, we can exploit these relationships algebraically to study the cycle indices, as we will see in the next section.

1.3. Algebra of species

It is often natural to describe a species in terms of combinations of other, simpler species—for example, 'a permutation is a set of cycles' or 'a rooted tree is a single vertex together with a (possibly empty) set of rooted trees'. Several combinatorial operations on species of structures are commonly used to represent these kinds of combinations; that they have direct analogues in the algebra of cycle indices is in some sense the conceptual justification of the theory. In particular, for species F and G, we will define species $F + G$, $F \cdot G$, $F \circ G$, $F^\bullet$, and F', and we will compute their cycle indices in terms of Z_F and Z_G. In what follows, we will not say explicitly what the effects of a given species operation are on bijections when those effects are obvious (as is usually the case).

Definition 1.3.1. For two species F and G, define their *sum* to be the species $F + G$ given by $\left(F + G\right)[A] = F[A] \sqcup G[A]$ (where $\sqcup$ denotes disjoint set union).

In other words, an $(F + G)$-structure is an F-structure *or* a G-structure. We use the disjoint union to avoid the complexities of dealing with cases where $F[A]$ and $G[A]$ overlap as sets.

THEOREM 1.3.2. *For species F and G, the cycle index of their sum is*

$$Z_{F+G} = Z_F + Z_G. \tag{5}$$

In the case that $F = G_1 + G_2$, we can simply invert the equation and write $F - G_2 = G_1$. However, we may instead wish to study the species $F - G$ without first writing F as a sum. In the spirit of the definition of species addition, we wish to define the species subtraction $F - G$ as the species of F-structures that 'are not' G-structures. For slightly more generality, we may apply the notions of Definition 1.1.4:

Definition 1.3.3. For two species F and G with a species embedding $\phi : G \to F$, define their *difference with respect to* ϕ to be the species $F \overset{\phi}{-} G$ given by $\left(F \overset{\phi}{-} G\right)[A] \coloneqq F[A] - \phi(G[A])$. When there is no ambiguity about the choice of embedding ϕ, especially in the case that G has a combinatorially natural embedding in F, we may instead simply write $F - G$ and call this species their *difference*.

For example, for $\mathcal{G}$ the species of graphs and $\mathfrak{a}$ the species of trees with the natural embedding, we have that $\mathcal{G} - \mathfrak{a}$ is the species of graphs with cycles.

We note also that species addition is associative and commutative (up to species isomorphism), and furthermore the empty species $\mathbf{0} : A \mapsto \varnothing$ is an additive identity, so species with addition form an abelian monoid. This can be completed to create the abelian group of *virtual species*, in which the subtraction $F - G$ of arbitrary species is defined; the two definitions in fact agree where our definition applies. We will not delve into the details of virtual species theory here, directing the reader instead to [**3**, §2.5].

Definition 1.3.4. For two species F and G, define their *product* to be the species $F \cdot G$ given by $(F \cdot G)[A] = \sum_{A=B \sqcup C} F[B] \times G[C]$.

In other words, an $(F \cdot G)$-structure is a partition of A into two sets B and C, an F-structure on B, and a G-structure on C. This definition is partially motivated by the following result on cycle indices:

THEOREM 1.3.5. *For species F and G, the cycle index of their product is*

$$Z_{F \cdot G} = Z_F \cdot Z_G. \tag{6}$$

Conceptually, the species product can be used to describe species that decompose uniquely into substructures of two specified species. For example, a permutation on a set A decomposes uniquely into a (possibly empty) set of fixed points and a derangement of their complement in A. Thus, $\mathcal{S} = \mathcal{E} \cdot \mathrm{Der}$ for $\mathcal{S}$ the species of permutations, $\mathcal{E}$ the species of sets, and Der the species of derangements.

We note also that species multiplication is commutative (up to species isomorphism) and distributes over addition, so the class of species with addition and multiplication forms a commutative semiring, with the species $\mathbf{1} : \begin{cases} \varnothing \mapsto \{\varnothing\} \\ A \neq \varnothing \mapsto \varnothing \end{cases}$ as a multiplicative identity; if addition is completed as previously described, the class of virtual species with addition and multiplication forms a true commutative ring.

In addition, the question of which species can be decomposed as sums and products without resorting to virtual species is one of great interest; the notions of *molecular* and *atomic* species are directly derived from such decompositions, and represent the beginnings of the systematic study of the structure of the class of species as a whole. Further details on this topic are presented in [**3**, §2.6].

Definition 1.3.6. For two species F and G with $G[\varnothing] = \varnothing$, define their *composition* to be the species $F \circ G$ given by $(F \circ G)[A] = \prod_{\pi \in P(A)} \left(F[\pi] \times \prod_{B \in \pi} G[B]\right)$ where $P(A)$ is the set of partitions of A.

In other words, the composition $F \circ G$ produces the species of F-structures of collections of G-structures. The definition is, again, motivated by a correspondence with a certain operation on cycle indices:

Definition 1.3.7. Let f and g be cycle indices. Then the *plethysm* $f \circ g$ is the cycle index

$$f \circ g = f\left(g\left(p_1, p_2, p_3, \ldots\right), g\left(p_2, p_4, p_6, \ldots\right), \ldots\right), \tag{7}$$

where $f(a, b, \ldots)$ denotes the cycle index f with a substituted for p_1, b substituted for p_2, and so on.

This definition is inherited directly from the theory of symmetric functions in infinitely many variables, where our p_i are basis elements, as previously discussed. This operation on cycle indices then corresponds exactly to species composition:

THEOREM 1.3.8. *For species F and G with $G[\varnothing] = \varnothing$, the cycle index of their plethysm is*

$$Z_{F \circ G} = Z_F \circ Z_G \tag{8}$$

where $\circ$ in the right-hand side is as in Eq. (7)*.*

Many combinatorial structures admit natural descriptions as compositions of species. For example, every graph admits a unique decomposition as a (possibly empty) set of (nonempty) connected graphs, so we have the species identity $\mathcal{G} = \mathcal{E} \circ \mathcal{G}^{\mathsf{C}}$ for $\mathcal{G}$ the species of graphs and $\mathcal{G}^{\mathsf{C}}$ the species of nonempty connected graphs.

Diligent readers may observe that the requirement that $G[\varnothing] = \varnothing$ in Definition 1.3.6 is in fact logically vacuous, since the given construction would simply ignore the $\varnothing$-structures. However, the formula in Theorem 1.3.8 fails to be well-defined for any Z_G with non-zero constant term (corresponding to species G with nonempty $G[\varnothing]$) unless Z_F has finite degree (corresponding to species F with support only in finitely many degrees). Consider the following example:

Example 1.3.9. Let $\mathcal{E}$ denote the species of sets, $\mathcal{E}_3$ its restriction to sets with three elements, $\mathbf{1}$ the species described above (which has one empty structure), and X the species of singletons (which has one order-1 structure). If $\mathcal{E}(\mathbf{1} + X)$ were well-defined, it would denote the species of 'partially-labeled sets'. However, for fixed cardinality n, there is an $\mathcal{E}(\mathbf{1} + X)$-structure on n labels *for each nonnegative k*—specifically, the set $[n]$ together with k unlabeled elements. Thus, there would be infinitely many structures of each cardinality for this 'species', so it is not in fact a species at all.

However, the situation for $\mathcal{E}_3(\mathbf{1}+X)$ is entirely different. A structure in this species is a 3-set, some of whose elements are labeled. There are only four possible such structures: $\{*,*,*\}$, $\{*,*,1\}$, $\{*,1,2\}$, and $\{1,2,3\}$, where $*$ denotes an unlabeled element and integers denote labeled elements. Moreover, by discarding the unlabeled elements, we can clearly see that $\mathcal{E}_3(\mathbf{1}+X) = \sum_{i=0}^{3} \mathcal{E}_i$.

In our setting, we will not use this alternative notion of composition, so we will not develop it formally here.

Several other binary operations on species are defined in the literature, including the Cartesian product $F \times G$, the functorial composition $F \square G$, and the inner plethysm $F \boxtimes G$ of **[24]**. We will not use these here. However, we do introduce two unary operations: $F^\bullet$ and F'.

Definition 1.3.10. For a species F, define its *species derivative* to be the species F' given by $F'[A] = F[A \cup \{*\}]$ for $*$ an element chosen not in A (say, the set A itself).

It is important to note that the label $*$ of an F'-structure is *distinguished* from the other labels; the automorphisms of the species F' cannot interchange $*$ with another label. Thus, species differentiation is appropriate for cases where we want to remove one 'position' in a structure. For example, for $\mathcal{L}$ the species of linear orders and $\mathcal{C}$ the species of cyclic orders, we have $\mathcal{L} = \mathcal{C}'$; a cyclic order on the set $A \cup \{*\}$ is naturally associated with the linear order on the set A produced by removing $*$. Terming this operation 'differentiation' is justified by its effect on cycle indices:

THEOREM 1.3.11. *For a species F, the cycle index of its derivative is given by*

$$Z_{F'}(p_1, p_2, \ldots) = \frac{\partial}{\partial p_1} Z_F(p_1, p_2, \ldots). \tag{9}$$

We note that we cannot in general recover Z_F from $Z_{F'}$, since there may be terms in Z_F which have no p_1-component (corresponding to F-structures which have no automorphisms with fixed points).

Finally, we introduce a variant of the species derivative which allows us to *label* the distinguished element $*$:

Definition 1.3.12. For a species F, define its *pointed species* to be the species $F^\bullet$ given by $F^\bullet[A] = F[A] \times A$ (that is, pairs of the form (f, a) where f is an F-structure on A and $a \in A$) with transport $F^\bullet[\sigma](f, a) = (F[\sigma](f), \sigma(a))$. We can also write $F^\bullet[A] = X \cdot F'$ for X the species of singletons.

In other words, an $F^\bullet[A]$-structure is an $F[A]$-structure with a distinguished element taken from the set A (as opposed to F', where the distinguished element is new). Thus, species pointing is appropriate for cases such as those of rooted trees: for $\mathfrak{a}$ the species of trees and $\mathcal{A}$ the species of rooted trees, we have $\mathcal{A} = \mathfrak{a}^\bullet$. Equation (9) leads directly to the following:

THEOREM 1.3.13. *For a species F, the cycle index of its corresponding pointed species is given by*

$$Z_{F^\bullet} = Z_X \cdot Z_{F'}. \tag{10}$$

Note that, again, we cannot in general recover Z_F from $Z_{F^\bullet}$, for the same reasons as in the case of $Z_{F'}$.

1.4. Multisort species

A species F as defined in Definition 1.1.1 is a functor $F : \mathbf{FinBij} \to \mathbf{FinSet}$; an F-structure in $F[A]$ takes its labels from the set A. The tool-set so produced is adequate to describe many classes of combinatorial structures. However, there is one particular structure type which it cannot effectively capture: the notion of distinct *sorts* of elements within a structure. Perhaps the most natural example of this is the case of k-colored graphs, where every vertex has one of k colors with the requirement that no pair of adjacent vertices shares a color. Automorphisms of such a graph must preserve the colorings of the vertices, which is not a natural restriction to impose in the calculation of the classical cycle index in Eq. (1). We thus incorporate the notion of sorts directly into a new definition:

Definition 1.4.1. For a fixed integer $k \geqslant 1$, define a *k-sort set* to be an ordered k-tuple of sets. Say that a k-sort set is *finite* if each component set is finite; in that case, its *k-sort cardinality* is the ordered tuple of its components' set cardinalities. Further, define a *k-sort function* to be an ordered k-tuple of set functions which acts componentwise on k-sort sets. For two k-sort sets U and V, a k-sort function σ is a *k-sort bijection* if each component is a set bijection. For k-sort sets of cardinality $(c_1, c_2, \ldots, c_k)$, denote by $\mathfrak{S}_{c_1,c_2,\ldots,c_k} = \mathfrak{S}_{c_1} \times \mathfrak{S}_{c_2} \times \cdots \times \mathfrak{S}_{c_k}$ the *k-sort symmetric group*, the elements of which are in natural bijection with k-sort bijections from a k-sort set to itself. Finally, denote by $\mathbf{FinBij}^k$ the category of finite k-sort sets with k-sort bijections.

We can then define an extension of species to the context of k-sort sets:

Definition 1.4.2. A *k-sort species* F is a functor $F : \mathbf{FinBij}^k \to \mathbf{FinBij}$ which associates to each k-sort set U a set $F[U]$ of *k-sort F-structures* and to each k-sort bijection $\sigma : U \to V$ a bijection $F[\sigma] : F[U] \to F[V]$.

Functorality once again imposes naturality conditions on these associations.

Just as in the theory of ordinary species, to each multisort species is associated a power series, its *cycle index*, which carries essential combinatorial data about the automorphism structure of the species. To keep track of the multiple sorts of labels, however, we require multiple sets of indeterminates. Where in ordinary cycle indices we simply used p_i for the ith indeterminate, we now use $p_i[j]$ for the ith indeterminate of the jth sort. In some contexts with small k, we will denote our sorts with letters (saying, for example, that we have 'X labels' and 'Y labels'), in which

case we will write $p_i[x]$, $p_i[y]$, and so forth. In natural analogy to Definition 1.2.1, the formula for the cycle index of a k-sort species F is given by

$$(11)\quad Z_F(p_1[1], p_2[1], \ldots; p_1[2], p_2[2], \ldots; \ldots; p_1[k], p_2[k], \ldots) = \sum_{\substack{n\geqslant 0 \\ a_1+a_2+\cdots+a_k=n}} \frac{1}{a_1!a_2!\ldots a_k!} \sum_{\sigma\in\mathfrak{S}_{a_1,a_2,\ldots,a_k}} \operatorname{fix}(F[\sigma])\, p_{[1]}^{\sigma_1} p_{[2]}^{\sigma_2} \ldots p_{[k]}^{\sigma_k}.$$

where by $p_{[i]}^{\sigma_i}$ we denote the product $\prod_j \left(p_j[i]\right)^{(\sigma_i)_j}$ where $(\sigma_i)_j$ is the number of j-cycles of σ_i.

The operations of addition and multiplication extend to the multisort context naturally. To make sense of differentiation and pointing, we need only specify a sort from which to draw the element or label which is marked; we then write F'^X and $F^{\bullet X}$ for the derivative and pointing respectively of F 'in the sort X', which is to say with its distinguished element drawn from that sort. When F is a 1-sort species and G a k-sort species, the construction of the k-sort species $F \circ G$ is natural; in other settings, we will not define a general notion of composition of multisort species.

1.5. Γ-species and quotient species

It is frequently the case that interesting combinatorial problems admit elegant descriptions in terms of quotients of a class of structures F under the action of a group Γ. In some cases, this group action will be *structural* in the sense that it commutes with permutations of labels in the species F, or, informally, that it is independent of the choice of labelings on each F-structure. In such a case, we may also say that Γ acts on 'unlabeled structures' of the class F.

Example 1.5.1. Let $\mathcal{G}$ denote the species of simple graphs. Let the group $\mathfrak{S}_2$ act on such graphs by letting the identity act trivially and letting the non-trivial element (12) send each graph to its complement (that is, by replacing each edge with a non-edge and each non-edge with an edge). This 'complementation action' is structural in the sense described previously.

We note that a group action is structural is exactly the condition that each $\gamma \in \Gamma$ acts by a species isomorphism $\gamma : F \to F$ in the sense of Definition 1.1.4.

We now incorporate such species-compatible actions into a new definition:

Definition 1.5.2. For Γ a group, a Γ-*species* F is a combinatorial species F together with an action of Γ on F-structures by species isomorphisms. Explicitly, for F a

Γ-species, the diagram

$$\begin{array}{ccccc} A & \xrightarrow{F} & F[A] & \xrightarrow{\gamma_A} & F[A] \\ \downarrow{\scriptstyle \sigma} & & \downarrow{\scriptstyle F[\sigma]} & & \downarrow{\scriptstyle F[\sigma]} \\ B & \xrightarrow{F} & F[B] & \xrightarrow{\gamma_B} & F[B] \end{array}$$

commutes for every $\gamma \in \Gamma$ and every set bijection $\sigma : A \to B$. (Note that commutativity of the left square is required for F to be a species at all.)

$\mathcal{G}$ is then a $\mathfrak{S}_2$-species with the action described in Example 1.5.1.

For such a Γ-species, of course, it is then meaningful to pass to the quotient under the action by Γ:

Definition 1.5.3. For F a Γ-species, define F/Γ, the *quotient species* of F under the action of Γ, to be the species of Γ-orbits of F-structures.

Example 1.5.4. Consider $\mathcal{G}$ as a $\mathfrak{S}_2$-species in light of the action defined in Example 1.5.1. The structures of the quotient species $\mathcal{G}/\mathfrak{S}_2$ are then pairs of complementary graphs. We may choose to interpret each such pair as representing a 2-partition of the set of vertex pairs of the complete graph (that is, of edges of the complete graph). More natural examples of quotient structures will present themselves in later chapters.

For each label set A, let $Q_\Gamma[A] : F[A] \to F/\Gamma[A]$ denote the map sending each F-structure over A to its quotient F/Γ-structure over A. Then $Q_\Gamma[A]$ is an injection for each A, and the requirement that Γ acts by natural transformations implies that the induced functor map $Q_\Gamma : F \to F/\Gamma$ is a natural transformation. Thus, the passage from F to F/Γ is a species cover in the sense of Definition 1.1.4.

A brief exposition of the notion of quotient species may be found in [**3**, §3.6], and a more thorough exposition (in French) in [**4**]. Our motivation, of course, is that combinatorial structures of a given class are often 'naturally' identified with orbits of structures of another, larger class under the action of some group. Our goal will be to compute the cycle index of the species F/Γ in terms of that of F and information about the Γ-action, so that enumerative data about the quotient species can be extracted.

As an intermediate step to the computation of the cycle index associated to this quotient species, we associate a cycle index to a Γ-species F that keeps track of the needed data about the Γ-action.

Definition 1.5.5. For a Γ-species F, define the Γ-cycle index Z_F^Γ as in **[14]**: for each $\gamma \in \Gamma$, let

$$Z_F^\Gamma(\gamma) = \sum_{n \geqslant 0} \frac{1}{n!} \sum_{\sigma \in \mathfrak{S}_n} \operatorname{fix}(\gamma \cdot F[\sigma])\, p_\sigma \tag{12}$$

with p_σ as in Eq. (1).

We will call such an object (formally a map from Γ to the ring $\mathbf{Q}[[p_1, p_2, \ldots]]$ of symmetric functions with rational coefficients in the p-basis) a Γ-*cycle index* even when it is not explicitly the Γ-cycle index of a Γ-species, and we will sometimes call $Z_F^\Gamma(\gamma)$ the "γ term of Z_F^Γ". So the coefficients in the power series count the fixed points of the *combined* action of a permutation and the group element γ. Note that, in particular, the classical ('ordinary') cycle index may be recovered as $Z_F = Z_F^\Gamma(e)$ for any Γ-species F.

The algebraic relationships between ordinary species and their cycle indices generally extend without modification to the Γ-species context, as long as appropriate allowances are made. The actions on cycle indices of Γ-species addition and multiplication are exactly as in the ordinary species case considered componentwise:

Definition 1.5.6. For two Γ-species F and G, the Γ-cycle index of their sum $F + G$ is given by

$$Z_{F+G}^\Gamma(\gamma) = Z_F^\Gamma(\gamma) + Z_G^\Gamma(\gamma) \tag{13}$$

and the Γ-cycle index of their product $F \cdot G$ is given by

$$Z_{F \cdot G}^\Gamma(\gamma) = Z_F^\Gamma(\gamma) \cdot Z_G^\Gamma(\gamma) \tag{14}$$

The action of composition, which in ordinary species corresponds to plethysm of cycle indices, can also be extended:

Definition 1.5.7. For two Γ-species F and G, define their *composition* to be the Γ-species $F \circ G$ with structures given by $(F \circ G)[A] = \prod_{\pi \in P(A)} \left(F[\pi] \times \prod_{B \in \pi} G[B]\right)$ where $P(A)$ is the set of partitions of A and where $\gamma \in \Gamma$ acts on a $(F \circ G)$-structure by acting on the F-structure and the G-structures independently.

The requirement in Definition 1.5.2 that the action of Γ commutes with transport implies that this is well-defined. Informally, for Γ-species F and G, we have defined the composition $F \circ G$ to be the Γ-species of F-structures of G-structures, where $\gamma \in \Gamma$ acts on an $(F \circ G)$-structure by acting independently on the F-structure and each of its associated G-structures. A formula similar to that Theorem 1.3.8 requires a definition of the plethysm of Γ-symmetric functions, here taken from **[14**, §3]:

Definition 1.5.8. For two Γ-cycle indices f and g, their *plethysm* $f \circ g$ is a Γ-cycle index defined by

$$(f \circ g)(\gamma) = f(\gamma)\left(g(\gamma)(p_1, p_2, p_3, \ldots), g(\gamma^2)(p_2, p_4, p_6, \ldots), \ldots\right). \tag{15}$$

This definition of Γ-cycle index plethysm is then indeed the correct operation to pair with the composition of Γ-species:

THEOREM 1.5.9 (Theorem 3.1, [**14**]). *If A and B are Γ-species and $B(\varnothing) = \varnothing$, then*

$$Z^{\Gamma}_{A \circ B} = Z^{\Gamma}_{A} \circ Z^{\Gamma}_{B}. \tag{16}$$

Thus, Γ-species admit the same sorts of 'nice' correspondences between structural descriptions (in terms of functorial algebra) and enumerative characterizations (in terms of cycle indices) that ordinary species do.

However, to make use of this theory for enumerative purposes, we also need to be able to pass from the Γ-cycle index of a Γ-species to the ordinary cycle index of its associated quotient species under the action of Γ. This will allow us to adopt a useful strategy: if we can characterize some difficult-to-enumerate combinatorial structure as quotients of more accessible structures, we will be able to apply the full force of species theory to the enumeration of the prequotient structures, *then* pass to the quotient when it is convenient. Exactly this approach will serve as the core of both of the following chapters.

Since we intend to enumerate orbits under a group action, we apply a generalization of Burnside's Lemma found in [**7**, Lemma 5]:

Lemma 1.5.10. *If Γ and Δ are finite groups and S a set with a $(\Gamma \times \Delta)$-action, for any $\delta \in \Delta$ the number of Γ-orbits fixed by δ is $\frac{1}{|\Gamma|}\sum_{\gamma \in \Gamma} \operatorname{fix}(\gamma, \delta)$.*

Recall from Eq. (1) that, to compute the cycle index of a species, we need to enumerate the fixed points of each $\sigma \in \mathfrak{S}_n$. However, to do this in the quotient species F/Γ is by definition to count the fixed Γ-orbits of σ in F under commuting actions of $\mathfrak{S}_n$ and Γ (that is, under an $(\mathfrak{S}_n \times \Gamma)$-action). Thus, Lemma 1.5.10 implies the following:

THEOREM 1.5.11. *For a Γ-species F, the ordinary cycle index of the quotient species F/Γ is given by*

$$Z_{F/\Gamma} = \overline{Z^{\Gamma}_{F}} := \frac{1}{|\Gamma|}\sum_{\gamma \in \Gamma} Z^{\Gamma}_{F}(\gamma) = \frac{1}{|\Gamma|}\sum_{\substack{n \geqslant 0 \\ \sigma \in \mathfrak{S}_n \\ \gamma \in \Gamma}} \frac{1}{n!}\left(\gamma \cdot F[\sigma]\right) p_{\sigma}. \tag{17}$$

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

Note that this same result on cycle indices is implicit in [**4**, §2.2.3]. With it, we can compute explicit enumerative data for a quotient species using cycle-index information of the original Γ-species with respect to the group action, as desired.

Recall from Theorems 1.2.2 and 1.2.3 that the exponential generating function $F(x)$ of labeled F-structures and the ordinary generating function $\tilde{F}(x)$ of unlabeled F-structures may both be computed from the cycle index Z_F of an ordinary species F by simple substitutions. In the Γ-species context, we may perform similar substitutions to derive analogous generating functions.

THEOREM 1.5.12. *The exponential generating function $F_\gamma(x)$ of labeled γ-invariant F-structures is*

$$F_\gamma(x) = Z_F^\Gamma(\gamma)(x, 0, 0, \ldots). \tag{18}$$

THEOREM 1.5.13. *The ordinary generating function $\tilde{F}_\gamma(x)$ of unlabeled γ-invariant F-structures is*

$$\tilde{F}_\gamma(x) = Z_F^\Gamma(\gamma)\left(x, x^2, x^3, \ldots\right). \tag{19}$$

These theorems follow directly from Eqs. (3) and (4), thinking of $F_\gamma(x)$ and $\widetilde{F_\gamma(x)}$ as enumerating the combinatorial class of F-structures which are invariant under γ.

Note that the notion of 'unlabeled γ-invariant F-structures' is always well-defined precisely because Definition 1.5.2 requires that the action of Γ commutes with transport of structures.

From these results and Theorem 1.5.11, we can the conclude:

THEOREM 1.5.14. *The exponential generating function $F(x)$ of labeled F/Γ-structures is*

$$F(x) = \frac{1}{|\Gamma|}\sum_{\gamma\in\Gamma} F_\gamma(x). \tag{20}$$

Similarly,

THEOREM 1.5.15. *The ordinary generating function $\tilde{F}(x)$ of unlabeled F/Γ-structures is*

$$\tilde{F}(x) = \frac{1}{|\Gamma|}\sum_{\gamma\in\Gamma} \tilde{F}_\gamma(x). \tag{21}$$

Note also that all of the above extends naturally into the multisort species context. We will use this extensively in Chapter 3. It also extends naturally to weighted contexts, but we will not apply this extension here.

CHAPTER 2

The species of bipartite blocks

2.1. Introduction

We first apply the theory of quotient species to the enumeration of bipartite blocks.

Definition 2.1.1. A *bicolored graph* is a graph Γ each vertex of which has been assigned one of two colors (here, black and white) such that each edge connects vertices of different colors. A *bipartite graph* (sometimes called *bicolorable*) is a graph Γ which admits such a coloring.

There is an extensive literature about bicolored and bipartite graphs, including enumerative results for bicolored graphs [**12**], bipartite graphs both allowing [**8**] and prohibiting [**13**] isolated points, and bipartite blocks [**11**]. However, this final enumeration was previously completed only in the labeled case. By considering the problem in light of the theory of Γ-species, we develop a more systematic understanding of the structural relationships between these various classes of graphs, which allows us to enumerate all of them in both labeled and unlabeled settings.

Throughout this chapter, we denote by $\mathcal{BC}$ the species of bicolored graphs and by $\mathcal{BP}$ the species of bipartite graphs. The prefix $\mathcal{C}$ will indicate the connected analogue of such a species.

We are motivated by the graph-theoretic fact that each *connected* bipartite graph may be identified with exactly two bicolored graphs which are color-dual. In other words, a connected bipartite graph is (by definition or by easy exercise, depending on your approach) an orbit of connected bicolored graphs under the action of $\mathfrak{S}_2$ where the nontrivial element τ reverses all vertex colors. We will hereafter treat all the various species of bicolored graphs as $\mathfrak{S}_2$-species with respect to this action and use the theory developed in Section 1.5 to pass to bipartite graphs.

Although the theory of multisort species presented in Section 1.4 is in general well-suited to the study of colored graphs, we will not need it here. The restrictions that vertex colorings place on automorphisms of bicolored graphs are simple enough that we can deal with them directly.

2.2. Bicolored graphs

We begin our investigation by directly computing the $\mathfrak{S}_2$-cycle index for the species $\mathcal{BC}$ of bicolored graphs with the color-reversing $\mathfrak{S}_2$-action described previously. We will then use various methods from the species algebra of Chapter 1 to pass to various other species.

2.2.1. Computing $Z^{\mathfrak{S}_2}_{\mathcal{BC}}(e)$. We construct the cycle index for the species $\mathcal{BC}$ of bicolored graphs in the classical way, which in light of our $\mathfrak{S}_2$-action will give $Z^{\mathfrak{S}_2}_{\mathcal{BC}}(e)$.

Recall the formula for the cycle index of a Γ-species in Eq. (12):

$$Z^{\Gamma}_{F}(\gamma) = \sum_{n \geqslant 0} \frac{1}{n!} \sum_{\sigma \in \mathfrak{S}_n} \operatorname{fix}(\gamma \cdot F[\sigma])\, p_\sigma.$$

Thus, for each $n > 0$ and each permutation $\pi \in \mathfrak{S}_n$, we must count bicolored graphs on $[n]$ for which π is a color-preserving automorphism. To simplify some future calculations, we omit empty graphs and define $\mathcal{BC}[\varnothing] = \varnothing$. We note that the *number* of such graphs in fact depends only on the cycle type $\lambda \vdash n$ of the permutation π, so we can use the cycle index formula in Eq. (2) interpreted as a Γ-cycle index identity.

Fix some $n \geqslant 0$ and let $\lambda \vdash n$. We wish to count bicolored graphs for which a chosen permutation π of cycle type λ is a color-preserving automorphism. Each cycle of the permutation must correspond to a monochromatic subset of the vertices, so we may construct graphs by drawing bicolored edges into a given colored vertex set. If we draw some particular bicolored edge, we must also draw every other edge in its orbit under π if π is to be an automorphism of the graph. Moreover, every bicolored graph for which π is an automorphism may be constructed in this way Therefore, we direct our attention first to counting these edge orbits for a fixed coloring; we will then count colorings with respect to these results to get our total cycle index.

Consider an edge connecting two cycles of lengths m and n; the length of its orbit under the permutation is $\operatorname{lcm}(m,n)$, so the number of such orbits of edges between these two cycles is $mn/\operatorname{lcm}(m,n) = \gcd(m,n)$. For an example in the case $m = 4, n = 2$, see Fig. 2.1. The number of orbits for a fixed coloring is then $\sum \gcd(m,n)$ where the sum is over the multiset of all cycle lengths m of white cycles and n of black cycles in the permutation π. We may then construct any possible graph fixed by our permutation by making a choice of a subset of these cycles to fill with edges, so the total number of such graphs is $\prod 2^{\gcd(m,n)}$ for a fixed coloring.

We now turn our attention to the possible colorings of the graph which are compatible with a permutation of specified cycle type λ. We split our partition into two subpartitions, writing $\lambda = \mu \cup \nu$, where partitions are treated as multisets

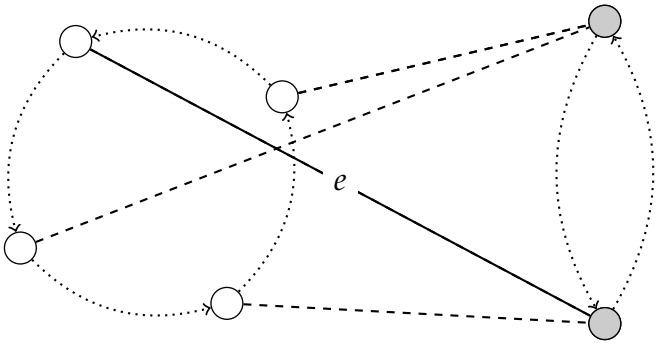


FIGURE 2.1. An edge e (solid) between two cycles of lengths 4 and 2 in a permutation and that edge's orbit (dashed)

and $\cup$ is the multiset union, and designate μ to represent the white cycles and ν the black. Then the total number of graphs fixed by such a permutation with a specified decomposition is

$$\operatorname{fix}(\mu,\nu) = \prod_{\substack{i\in\mu\\ j\in\nu}} 2^{\gcd(i,j)}$$

where the product is over the elements of μ and λ taken as multisets. However, since μ and ν represent white and black cycles respectively, it is important to distinguish *which* cycles of λ are taken into each. The λ_i i-cycles of λ can be distributed into μ and ν in $\binom{\lambda_i}{\mu_i} = \lambda_i!/(\mu_i!\nu_i!)$ ways, so in total there are $\prod_i \lambda_i!/(\mu_i!\nu_i!) = z_\lambda/(z_\mu z_\nu)$ decompositions. Thus,

$$\operatorname{fix}(\lambda) = \frac{z_\lambda}{z_\mu z_\nu}\operatorname{fix}(\mu,\nu) = \sum_{\mu\cup\nu=\lambda} \frac{z_\lambda}{z_\mu z_\nu} \prod_{\substack{i\in\mu\\ j\in\nu}} 2^{\gcd(i,j)}.$$

Therefore we conclude:

THEOREM 2.2.1.

$$Z^{\mathfrak{S}_2}_{\mathcal{BC}}(e) = \sum_{n>0} \sum_{\substack{\mu,\nu\\ \mu\cup\nu\vdash n}} \frac{p_{\mu\cup\nu}}{z_\mu z_\nu} \prod_{i,j} 2^{\gcd(\mu_i,\nu_j)} \tag{22}$$

Explicit formulas for the generating function for unlabeled bicolored graphs were obtained in **[12]** using conventional Pólya-theoretic methods. Conceptually, this enumeration in fact largely mirrors our own. Harary uses the algebra of the classical cycle index of the 'line group[1]' of the complete bicolored graph of which any given bicolored graph is a spanning subgraph. He then enumerates orbits of

[1]The *line group* of a graph is the group of permutations of edges induced by permutations of vertices.

edges under these groups using the Pólya enumeration theorem. This is clearly analogous to our procedure, which enumerates the orbits of edges under each specific permutation of vertices.

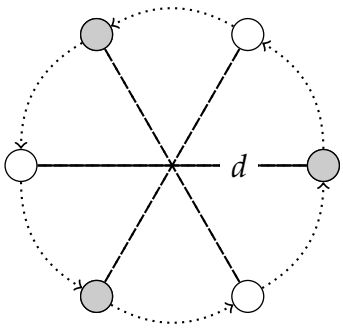
2.2.2. Calculating $Z_{\mathcal{BC}}^{\mathfrak{S}_2}(\tau)$. Recall that the nontrivial element of $\tau \in \mathfrak{S}_2$ acts on bicolored graphs by reversing all colors.

We again consider the cycles in the vertex set $[n]$ induced by a permutation $\pi \in \mathfrak{S}_n$ and use the partition λ corresponding to the cycle type of π for bookkeeping. We then wish to count bicolored graphs on $[n]$ for which $\tau \cdot \pi$ is an automorphism, which is to say that π itself is a color-*reversing* automorphism. Once again, the number of bicolored graphs for which π is a color-reversing automorphism is in fact dependent only on the cycle type λ. Each cycle of vertices must be color-alternating and hence of even length, so our partition λ must have only even parts. Once this condition is satisfied, edges may be drawn either within a single cycle or between two cycles, and as before if we draw in any edge we must draw in its entire orbit under π (since π is to be an automorphism of the underlying graph). Moreover, all graphs for which π is a color-reversing automorphism and with a fixed coloring may be constructed in this way, so it suffices to count such edge orbits and then consider how colorings may be assigned.

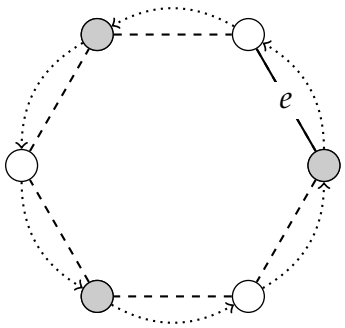
Consider a cycle of length $2n$; we hereafter describe such a cycle as having *semilength* n. There are exactly n^2 possible white-black edges in such a cycle. If n is odd, diametrically opposed vertices have opposite colors, so we can have an edge of length $l = n$ (in the sense of connecting two vertices which are l steps apart in the cycle), and in such a case the orbit length is exactly n and there is exactly one orbit. See Fig. 2.2a for an example of this case. However, if n is odd but $l \neq n$, the orbit length is $2n$, so the number of such orbits is $\frac{n^2-n}{2n}$. Hence, the total number of orbits for n odd is $\frac{n^2+n}{2n} = \left\lceil \frac{n}{2} \right\rceil$. Similarly, if n is even, all orbits are of length $2n$, so the total number of orbits is $\frac{n^2}{2n} = \frac{n}{2} = \left\lceil \frac{n}{2} \right\rceil$ also. See Fig. 2.2b for an example of each of these cases.

Now consider an edge to be drawn between two cycles of semilengths m and n. The total number of possible white-black edges is $2mn$, each of which has an orbit length of $\operatorname{lcm}(2m, 2n) = 2\operatorname{lcm}(m, n)$. Hence, the total number of orbits is $2mn / (2\operatorname{lcm}(m, n)) = \gcd(m, n)$.

All together, then, the number of orbits for a fixed coloring of a permutation of cycle type 2λ (denoting the partition obtained by doubling every part of λ) is $\sum_i \left\lceil \frac{\lambda_i}{2} \right\rceil + \sum_{i<j} \gcd(\lambda_i, \lambda_j)$. All valid bicolored graphs for a fixed coloring for which π is a color-preserving automorphism may be obtained uniquely by making some choice of a subset of this collection of orbits, just as in Section 2.2.1. Thus,



(A) A diameter d ($l = 3$)



(B) A non-diameter e ($l = 1$)

FIGURE 2.2. Both types of intra-cycle edges and their orbits on a typical color-alternating 6-cycle

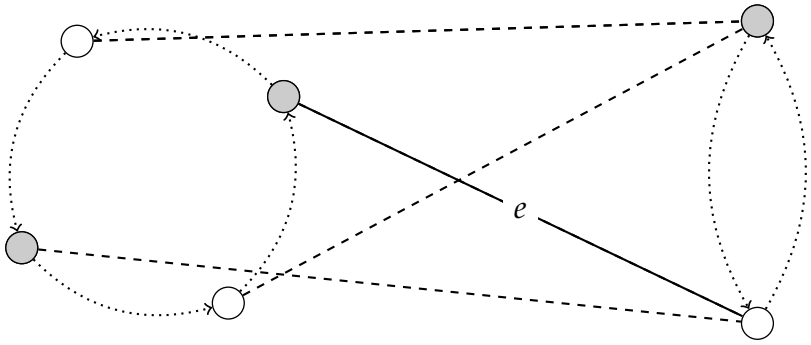


FIGURE 2.3. An edge e and its orbit between color-alternating cycles of semilengths 2 and 1

the total number of possible graphs for a given vertex coloring is

$$\prod_i 2^{\left\lceil \frac{\lambda_i}{2} \right\rceil} \prod_{i<j} 2^{\gcd(\lambda_i, \lambda_j)},$$

which we note is independent of the choice of coloring. For a partition 2λ with $l(\lambda)$ cycles, there are then $2^{l(\lambda)}$ colorings compatible with our requirement that each cycle is color-alternating, which we multiply by the previous to obtain the total number of graphs for all permutations π with cycle type 2λ. Therefore we conclude:

THEOREM 2.2.2.

$$Z^{\mathfrak{S}_2}_{\mathcal{BC}}(\tau) = \sum_{\substack{n>0 \\ n \text{ even}}} \sum_{\lambda \vdash \frac{n}{2}} 2^{l(\lambda)} \frac{p_{2\lambda}}{z_{2\lambda}} \prod_i 2^{\left\lceil \frac{\lambda_i}{2} \right\rceil} \prod_{i<j} 2^{\gcd(\lambda_i, \lambda_j)} \tag{23}$$

2.3. Connected bicolored graphs

As noted in the introduction of this section, we may pass from bicolored to bipartite graphs by taking a quotient under the color-reversing action of $\mathfrak{S}_2$ only in the connected case. Thus, we must pass from the species $\mathcal{BC}$ to the species $\mathcal{CBC}$ of connected bicolored graphs to continue. It is a standard principle of graph enumeration that a graph may be decomposed uniquely into (and thus species-theoretically identified with) the set of its connected components. We must, of course, require that the component structures are nonempty to ensure that the construction is well-defined, as discussed in Section 1.3. This same relationship holds in the case of bicolored graphs. Thus, the species $\mathcal{BC}$ of nonempty bicolored graphs is the composition of the species $\mathcal{CBC}$ of nonempty connected bicolored graphs into the species $\mathcal{E}^+ = \mathcal{E} - 1$ of nonempty sets:

$$\mathcal{BC} = \mathcal{E}^+ \circ \mathcal{CBC} \tag{24}$$

Reversing the colors of a bicolored graph is done simply by reversing the colors of each of its connected components independently; thus, once we trivially extend the species $\mathcal{E}^+$ to an $\mathfrak{S}_2$-species by applying the trivial action, Eq. (24) holds as an identity of $\mathfrak{S}_2$-species for the color-reversing $\mathfrak{S}_2$-action described previously.

To use the decomposition in Eq. (24) to derive the $\mathfrak{S}_2$-cycle index for $\mathcal{CBC}$, we must invert the $\mathfrak{S}_2$-species composition into $\mathcal{E}^+$. In the context of the theory of virtual species, this is possible; we write $\mathrm{Con} := (\mathcal{E} - 1)^{\langle -1 \rangle}$ to denote this virtual species. We can derive from [**3**, §2.5, eq. (58c)] that its cycle index is

$$Z_{\mathrm{Con}} = \sum_{k \geqslant 1} \frac{\mu(k)}{k} \log(1 + p_k) \tag{25}$$

where μ is the Möbius function. We can then rewrite Eq. (24) as

$$\mathcal{CBC} = \mathrm{Con} \circ \mathcal{BC}$$

It then follows immediately from Theorem 1.5.9 that

THEOREM 2.3.1.

$$Z^{\mathfrak{S}_2}_{\mathcal{CBC}} = Z_{\mathrm{Con}} \circ Z^{\mathfrak{S}_2}_{\mathcal{BC}} \tag{26}$$

2.4. Bipartite graphs

As we previously observed, connected bipartite graphs are naturally identified with orbits of connected bicolored graphs under the color-reversing action of $\mathfrak{S}_2$. Thus,

$$\mathcal{CBP} = {}^{\mathcal{CBC}} \big/ \mathfrak{S}_2.$$

By application of Theorem 1.5.11, we can then directly compute the cycle index of $\mathcal{CBP}$ in terms of previous results:

THEOREM 2.4.1.

$$Z_{\mathcal{CBP}} = \overline{Z^{\mathfrak{S}_2}_{\mathcal{CBC}}} = \frac{1}{2}\left(Z^{\mathfrak{S}_2}_{\mathcal{CBC}}(e) + Z^{\mathfrak{S}_2}_{\mathcal{CBC}}(\tau)\right). \tag{27}$$

Finally, to reach a result for the general bipartite case, we return to the graph-theoretic composition relationship previously considered in Section 2.3:

$$\mathcal{BP} = \mathcal{E} \circ \mathcal{CBP}.$$

This time, we need not invert the composition, so the cycle-index calculation is simple:

THEOREM 2.4.2.

$$Z_{\mathcal{BP}} = Z_{\mathcal{E}} \circ Z_{\mathcal{CBP}}. \tag{28}$$

A generating function for labeled bipartite graphs was obtained first in [**13**] and later in [**8**]; the latter uses Pólya-theoretic methods to calculate the cycle index of what in modern terminology would be the species of edge-labeled complete bipartite graphs.

2.5. Nonseparable graphs

We now turn our attention to the notions of block decomposition and nonseparable graphs. A graph is said to be *nonseparable* if it is vertex-2-connected (that is, if there exists no vertex whose removal disconnects the graph); every connected graph then has a canonical 'decomposition'[2] into maximal nonseparable subgraphs, often shortened to *blocks*. In the spirit of our previous notation, we we will denote by $\mathcal{NBP}$ the species of nonseparable bipartite graphs, our object of study.

The basic principles of block enumeration in terms of automorphisms and cycle indices of permutation groups were first identified and exploited in [**22**]. In [**3**, §4.2], a theory relating a specified species B of nonseparable graphs to the species C_B of connected graphs whose blocks are in B is developed using similar principles. It is apparent that the class of nonseparable bipartite graphs is itself exactly the class of blocks that occur in block decompositions of connected bipartite graphs; hence, we apply that theory here to study the species $\mathcal{NBP}$. From [**3**, eq. 4.2.27] we obtain

THEOREM 2.5.1.

$$\mathcal{NBP} = \mathcal{CBP}\left(\mathcal{CBP}^{\bullet\langle -1\rangle}\right) + X \cdot \mathcal{NBP}' - X, \tag{29a}$$

[2]Note that this decomposition does not actually partition the vertices, since many blocks may share a single cut-point, a detail which significantly complicates but does not entirely preclude species-theoretic analysis.

where by [**3**, 4.2.26(a)] *we have*

$$\mathcal{NBP}' = \mathrm{Con}\left(\frac{X}{\mathcal{CBP}^{\bullet\langle -1\rangle}}\right). \tag{29b}$$

We have already calculated the cycle index for the species $\mathcal{CBP}$, so the calculation of the cycle index of $\mathcal{NBP}$ is now simply a matter of algebraic expansion.

A generating function for labeled bipartite blocks was given in [**11**], where their analogue of Eq. (29) for the labeled exponential generating function for blocks comes from [**5**]. However, we could locate no corresponding unlabeled enumeration in the literature. The numbers of labeled and unlabeled nonseparable bipartite graphs for $n \leqslant 10$ as calculated using our method are given in Table 1.

CHAPTER 3

The species of k-trees

3.1. Introduction

3.1.1. 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 $k \in \mathbf{N}$) may be defined recursively:

Definition 3.1.1. 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 [**10**]; vertex-labeled k-trees were quickly enumerated in the following year in both [**19**] 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 [**20**], for plane 2-trees in [**21**], and for unlabeled 2-trees in [**10**] and [**9**]. In 2001, the theory of species was brought to bear on 2-trees in [**6**], 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 3.1.2. 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 3.1.1 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. 3.1), while 3-trees are formed by gluing tetrahedra together along their triangular faces without forming loops of tetrahedra.

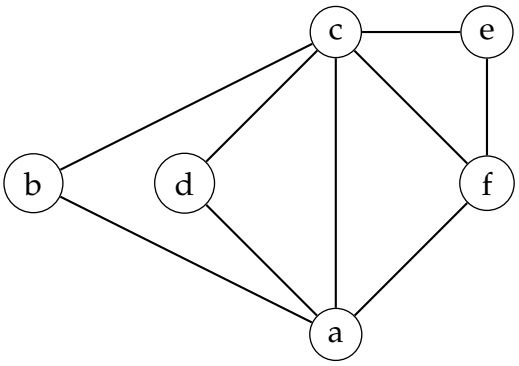


FIGURE 3.1. A (vertex-labeled) 2-tree

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. 3.2).

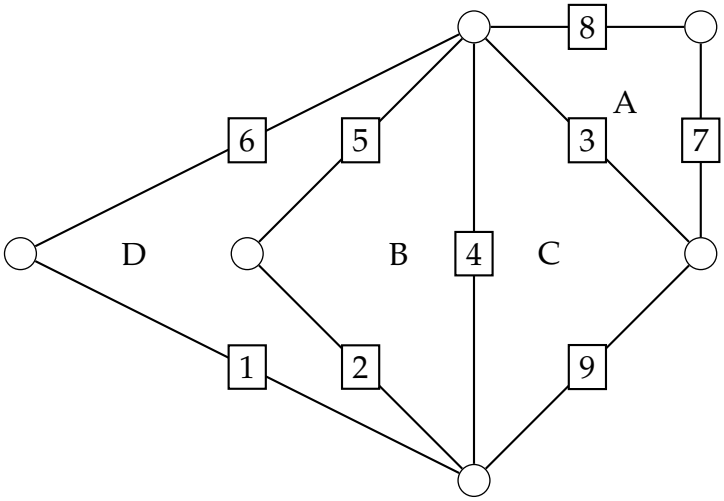


FIGURE 3.2. A (hedron-and-front–labeled) 2-tree

3.2. The dissymmetry theorem for *k*-trees

Studies of tree-like structures—especially those explicitly informed by the theory of species—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 $\mathcal{A} = \mathfrak{a}^\bullet$ 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.

A similar theorem holds for k-trees.

THEOREM 3.2.1. *The species $\mathfrak{a}_k^X$ and $\mathfrak{a}_k^Y$ of k-trees rooted at hedra and fronts respectively, $\mathfrak{a}_k^{XY}$ of k-trees rooted at a hedron with a designated front, and $\mathfrak{a}_k$ of unrooted k-trees are related by the equation*

$$\mathfrak{a}_k^X + \mathfrak{a}_k^Y = \mathfrak{a}_k + \mathfrak{a}_k^{XY} \tag{30}$$

as an isomorphism of species.

PROOF. We give a bijective, natural map from $\left(\mathfrak{a}_k^X + \mathfrak{a}_k^Y\right)$-structures on the left side to $\left(\mathfrak{a}_k + \mathfrak{a}_k^{XY}\right)$-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 [**16**], 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 $\left(\mathfrak{a}_k^X + \mathfrak{a}_k^Y\right)$-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 $\mathfrak{a}_k$ on the right-hand side. This map is a natural bijection from its preimage, the set of k-trees rooted at their centers, to $\mathfrak{a}_k$, 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 $\mathfrak{a}_k^{XY}$ of k-trees rooted at an adjacent hedron-front pair.

The combination of these two maps then gives the desired isomorphism of species in Eq. (30). □

In general we will reformulate the dissymmetry theorem as follows:

Corollary 3.2.2. *For the various forms of the species $\mathfrak{a}_k$ as above, we have*

$$\mathfrak{a}_k = \mathfrak{a}_k^X + \mathfrak{a}_k^Y - \mathfrak{a}_k^{XY}. \tag{31}$$

as an isomorphism of ordinary species.

This species subtraction is well-defined in the sense of Definition 1.3.3, since the species $\mathfrak{a}_k^{XY}$ embeds in the species $\mathfrak{a}_k^X + \mathfrak{a}_k^Y$ by the centering map described in the proof of Theorem 3.2.1. Essentially, Eq. (31) identifies each unrooted k-tree with itself rooted at its center simplex.

Theorem 3.2.1 and the consequent Eq. (31) 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.

3.3. Coherently-oriented k-trees

3.3.1. Symmetry-breaking. In the case of the species $\mathcal{A} = \mathfrak{a}_1^\bullet$ of rooted trees, we may obtain a simple recursive functional equation [**3**, §1, eq. (9)]:

$$\mathcal{A} = X \cdot \mathcal{E}(\mathcal{A}). \tag{32}$$

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 3.3.1. 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 3.3.2. *Let T be a coherently-oriented 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 3.3.3. 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. 3.3 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 $\vec{\mathfrak{a}}_k$ 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 $\vec{\mathfrak{a}}_k$ whose orbits are generic k-trees we can use the theory of quotient species to

extract the generic species $\mathfrak{a}_k$. First, however, we describe an encoding procedure which will make future work more convenient.

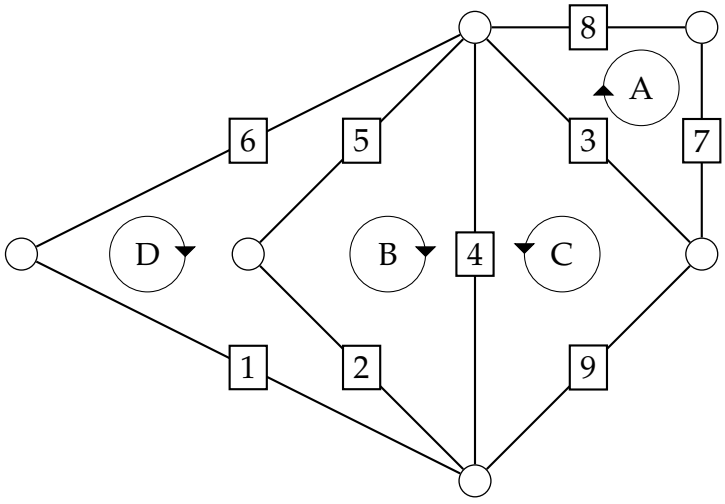


FIGURE 3.3. A coherently-oriented 2-tree

3.3.2. 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 hedra 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 $(\mathcal{C}_{k+1}, \mathcal{E})$-enriched bicolored tree is a bicolored tree each black vertex of which carries a $\mathcal{C}_{k+1}$-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 $\mathcal{CT}_k$ the species of such k-coding trees.

We now define a map $\beta : \vec{\mathfrak{a}}_k[n] \to \mathcal{CT}_k[n]$. 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 $\mathcal{C}_{k+1}$-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. 3.4, which shows the 2-coding tree associated to the coherently-oriented 2-tree of Fig. 3.3. Note that, for clarity, we have rendered the black vertices (corresponding to hedra) with squares.

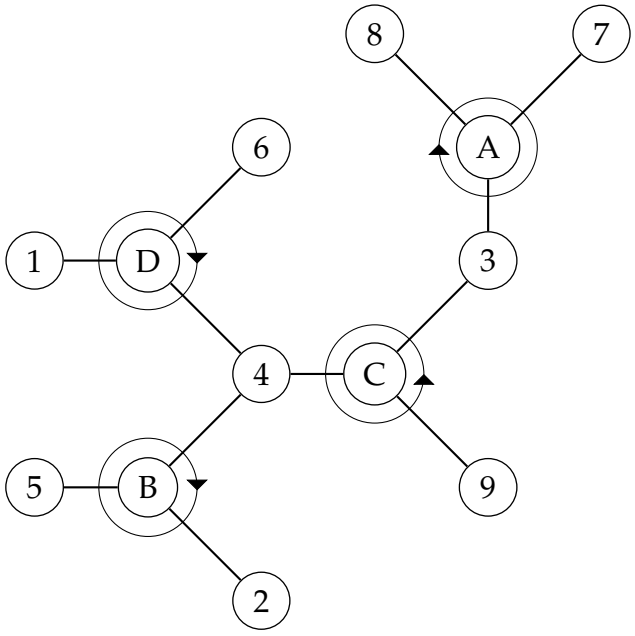


FIGURE 3.4. A $(\mathcal{C}_{k+1}, \mathcal{E})$-enriched bicolored tree encoding a coherently-oriented 2-tree

THEOREM 3.3.4. *The map β induces an isomorphism of species $\vec{\mathfrak{a}}_k \simeq \mathcal{CT}_k$.*

PROOF. It is clear that β 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 β induces a species isomorphism, then, we need only show that β is a surjection onto $\mathcal{CT}_k[n]$ for each n. Throughout, we will say 'F and G have contact of order n' when the restrictions $F_{\leqslant n}$ and $G_{\leqslant n}$ 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 β is clearly a natural bijection between these two sets. Thus, the species $\vec{\mathfrak{a}}_k$ of coherently-oriented k-trees and $\mathcal{CT}_k$ of k-coding trees have contact of order 1.

Now, by way of induction, suppose $\vec{\mathfrak{a}}_k$ and $\mathcal{CT}_k$ have contact of order $n \geqslant 1$. 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 $\vec{\mathfrak{a}}_k$ and $\mathcal{CT}_k$ have contact of order $n+1$. □

Thus, $\vec{\mathfrak{a}}_k$ and $\mathcal{CT}_k$ 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 $\mathcal{CT}_k$, then lift our results to the k-tree context.

3.3.3. Functional decomposition of k-coding trees. With the encoding $\beta : \vec{\mathfrak{a}}_k \to \mathcal{CT}_k$, 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 $\mathcal{CT}_k$ of k-coding trees first, then use dissymmetry to apply the results to unrooted enumeration.

Let $\mathcal{CT}_k^X$ denote the species of k-coding trees rooted at black vertices, $\mathcal{CT}_k^Y$ denote the species of k-coding trees rooted at white vertices, and $\mathcal{CT}_k^{XY}$ denote the species of k-coding trees rooted at edges (that is, at adjacent black-white pairs). By construction, a $\mathcal{CT}_k^X$-structure consists of a single X-label and a cyclically-ordered $(k+1)$-set of $\mathcal{CT}_k^Y$-structures. See Fig. 3.5 for an example of this construction.

Similarly, a $\mathcal{CT}_k^Y$-structure essentially consists of a single Y-label and a (possibly empty) set of $\mathcal{CT}_k^X$-structures, but with some modification. Every white neighbor of the black root of a $\mathcal{CT}_k^X$-structure is labeled in the construction above, but the white parent of a $\mathcal{CT}_k^X$-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 $\mathcal{CT}_k^Y$-structures. Thus, a $\mathcal{CT}_k^Y$-structure consists of a Y-label and a set of pairs of an X label and an $\mathcal{L}_k$-structure of $\mathcal{CT}_k^Y$-structures. 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 $\mathcal{CT}_k^{XY}$-structure is simply an $X \cdot \mathcal{L}_k(\mathcal{CT}_k^Y)$-structure as described above (corresponding to the black vertex) together with a $\mathcal{CT}_k^Y$-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

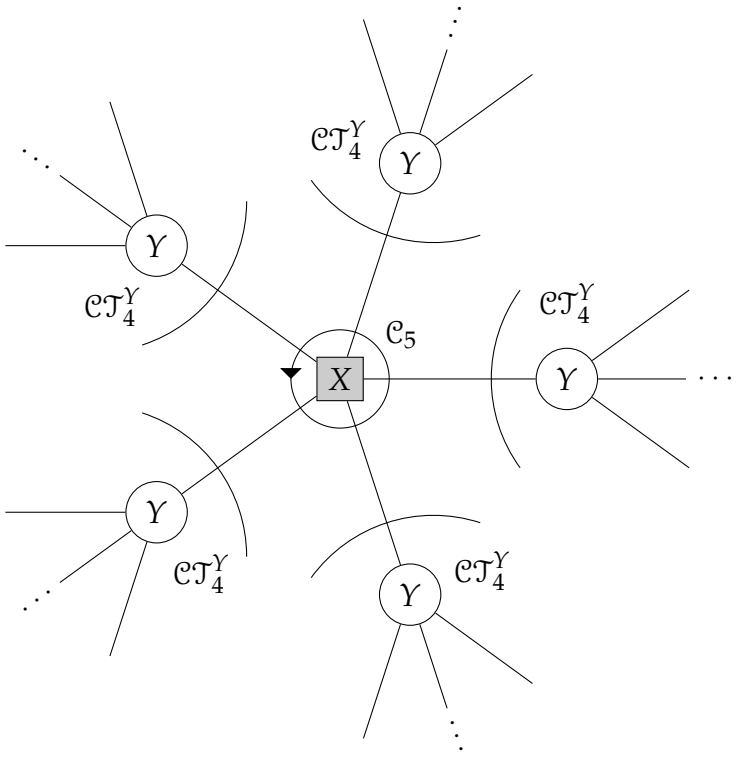


FIGURE 3.5. An example $\mathcal{CT}_4^X$-structure, rooted at the X-vertex.

representing a $\mathcal{CT}_k^{XY}$-structure instead as an $X \cdot \mathcal{L}_{k+1}\big(\mathcal{CT}_k^Y\big)$-structure. See Fig. 3.6 for an example of this construction.

The various species of rooted k-coding trees are therefore related by a system of functional equations:

Observation 3.3.5. *For the (ordinary) species $\mathcal{CT}_k^X$ of X-rooted k-coding trees, $\mathcal{CT}_k^Y$ of Y-rooted k-coding trees, and $\mathcal{CT}_k^{XY}$ of edge-rooted k-coding trees, we have the functional relationships*

$$\mathcal{CT}_k^X = X \cdot \mathcal{C}_{k+1}\big(\mathcal{CT}_k^Y\big) \tag{33a}$$

$$\mathcal{CT}_k^Y = Y \cdot \mathcal{E}\left(X \cdot \mathcal{L}_k\big(\mathcal{CT}_k^Y\big)\right) \tag{33b}$$

$$\mathcal{CT}_k^{XY} = \mathcal{CT}_k^Y \cdot X \cdot \mathcal{L}_k\big(\mathcal{CT}_k^Y\big) = X \cdot \mathcal{L}_{k+1}\big(\mathcal{CT}_k^Y\big) \tag{33c}$$

as isomorphisms of ordinary species.

However, a recursive characterization of the various ordinary 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.

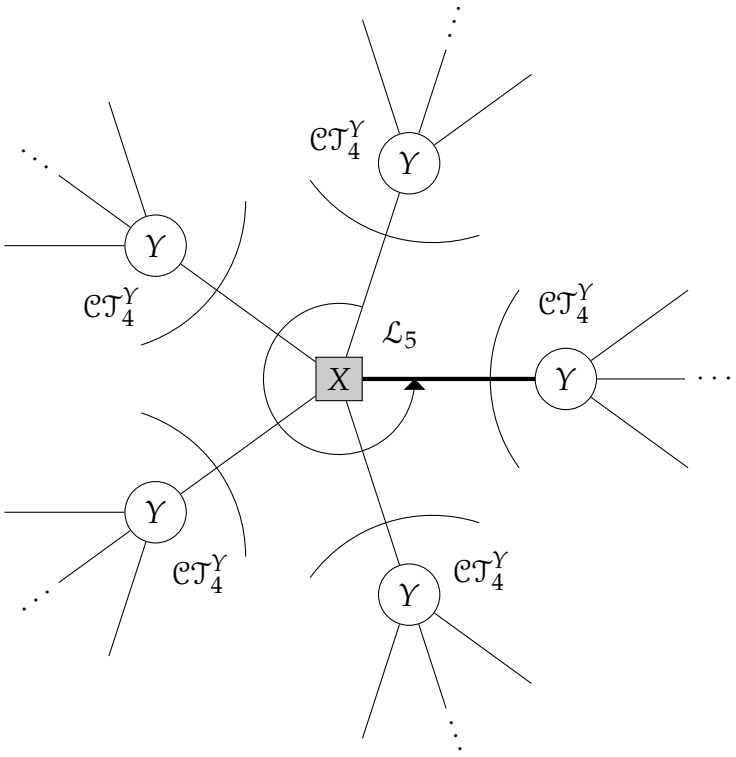


FIGURE 3.6. An example $\mathcal{CT}_4^{XY}$-structure, rooted at the X-vertex and the thick edge adjoining it.

3.4. Generic k-trees

To remove the additional structure of coherent orientation imposed on k-trees before their conversion to k-coding trees, we now apply the theory of Γ-species developed in Section 1.5. In **[6]**, the orientation-reversing action of $\mathfrak{S}_2$ on $\mathrm{Cyc}_{[3]}$ 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 $\mathfrak{S}_k$. Unfortunately:

Proposition 3.4.1. *For $k \geqslant 3$, no transitive action of any group on the set $\mathrm{Cyc}_{[k+1]}$ of cyclic orders on $[k+1]$ commutes with the action of $\mathfrak{S}_{k+1}$ that permutes labels.*

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

we have that

$$g \cdot c = g \cdot ccc^{-1} = c\,(g \cdot c)\,c^{-1} \tag{34}$$

and so c and $g \cdot c$ commute. Thus, c commutes with every element of its orbit under the action of G. But, for $k \geqslant 3$, not all elements of $\mathrm{Cyc}_{[k+1]}$ 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. Accordingly, we cannot simply apply the results of Section 3.3.3 to compute a Γ-species $\mathcal{CT}_k$ with respect to some hypothetical action of a group Γ whose orbits correspond to generic k-trees. 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 $\mathfrak{S}_{k+1}$.

3.4.1. 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 $\mathfrak{S}_{k+1}$ 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 $\mathfrak{S}_{k+1}$ 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 $\sigma \in \mathfrak{S}_{k+1}$ 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 σ. 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 σ, orbits under all of $\mathfrak{S}$ 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 σ 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 $\sigma \in \mathfrak{S}_{k+1}$ 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 σ' 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 σ, and then passing back. We can express this procedure in formulaic terms:

THEOREM 3.4.2. *If a permutation $\sigma \in \mathfrak{S}_{k+1}$ is applied to 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 ith white child of v (with respect to the linear ordering induced by the orientation) to preserve the mirror relation is $\rho_i(\sigma)$, where ρ_i is the map given by*

$$\rho_i(\sigma) : a \mapsto \sigma(i+a) - \sigma(i) \tag{35}$$

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 child of v' which is ath in the linear order induced by the original orientation around v'. It is mirror to the white child 3 of v which is $(i+a)$th in the linear order induced by the original orientation around v. After the action of σ 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. 3.7.

As an aside, we note that, although the construction ρ depends on k, the value of k will be fixed in any given context, so we suppress it in the notation.

Any σ which is to be applied to a non-root black vertex v must of course fix $k+1$. We let $\Delta : \mathfrak{S}_k \to \mathfrak{S}_{k+1}$ denote the obvious embedding; then the image of Δ is exactly the set of $\sigma \in \mathfrak{S}_{k+1}$ which fix $k+1$. We then have an action of $\mathfrak{S}_k$ on non-root black vertices induced by Δ. (Equivalently, we can think of $\mathfrak{S}_k$ as the subgroup of $\mathfrak{S}_{k+1}$ of permutations fixing $k+1$, but the explicit notation Δ will be of use in later formulas.)

In light of Observation 3.3.5, we now wish to adapt these ideas into explicit $\mathfrak{S}_k$- and $\mathfrak{S}_{k+1}$-actions on $\mathcal{CT}_k^X$, $\mathcal{CT}_k^Y$, and $\mathcal{CT}_k^{XY}$ 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 (as $\Delta(\sigma)$) on each of the black vertices immediately adjacent to the root and then applying ρ-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 σ) we can simply ignore the black vertex in the root.

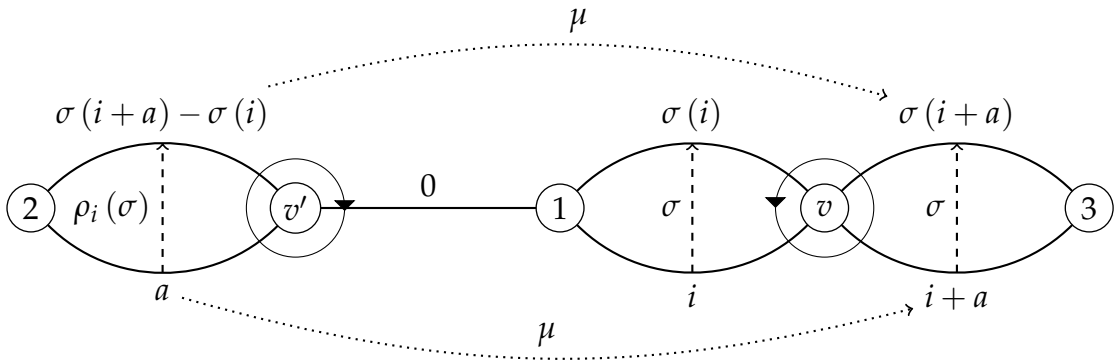


FIGURE 3.7. Application of a permutation σ to the orientation of a non-root black vertex v. The vertices 2 and 3 are mirror in the original orientation (lower set of edges), as shown by the arrows μ, so we must preserve this mirror relation when we apply σ. The permutation σ moves 3 from the $(i+a)$th place to the $\sigma(i+a)$th, so $\rho_i(\sigma)$ must carry 2 from the ath place to the $(\sigma(i+a)-\sigma(i))$th.

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.

3.4.2. *k*-trees as quotients. Since these actions are label-independent, we may now treat $\mathcal{CT}_k^Y$ and $\mathcal{CT}_k^{XY}$ as $\mathfrak{S}_k$-species and $\mathcal{CT}_k^{XY}$ as an $\mathfrak{S}_{k+1}$-species. 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. As a result of the above results, we can then relate the rooted Γ-species forms of $\mathcal{CT}_k$ to the various ordinary species forms of generic rooted k-trees in Theorem 3.2.1:

THEOREM 3.4.3. *For the various rooted forms of the ordinary species $\mathfrak{a}_k$ as in Theorem 3.2.1 and the various rooted Γ-species forms of $\mathcal{CT}_k$ as in Observation 3.3.5 as $\mathfrak{S}_k$- and*

$\mathfrak{S}_{k+1}$*-species, we have*

$$\mathfrak{a}_k^Y = {}^{\mathcal{CT}_k^Y}\!\!\diagup_{\mathfrak{S}_k} \tag{36a}$$

$$\mathfrak{a}_k^{XY} = {}^{\mathcal{CT}_k^{XY}}\!\!\diagup_{\mathfrak{S}_k} \tag{36b}$$

$$\mathfrak{a}_k^X = {}^{\mathcal{CT}_k^{XY}}\!\!\diagup_{\mathfrak{S}_{k+1}} \tag{36c}$$

as isomorphisms of ordinary species, where $\mathcal{CT}_k^{XY}$ *is an* $\mathfrak{S}_k$*-species in Eq.* (36b) *and an* $\mathfrak{S}_{k+1}$*-species in Eq.* (36c)*.*

As a result, we have explicit characterizations of all the rooted components of the original dissymmetry theorem, Theorem 3.2.1. To compute the cycle indices of these components (and thus the cycle index of $\mathfrak{a}_k$ itself), we need only compute the cycle indices of the various rooted $\mathcal{CT}_k$ species, which we will do using a combination of the functional equations in Eq. (33) and explicit consideration of automorphisms.

3.5. Automorphisms and cycle indices

3.5.1. ***k*-coding trees: $\mathcal{CT}_k^Y$ and $\mathcal{CT}_k^{XY}$.** Corollary 3.2.2 of the dissymmetry theorem for k-trees has a direct analogue in terms of cycle indices:

THEOREM 3.5.1. *For the various forms of the species* $\mathfrak{a}_k$ *as in Section 3.2, we have*

$$Z_{\mathfrak{a}_k} = Z_{\mathfrak{a}_k^X} + Z_{\mathfrak{a}_k^Y} - Z_{\mathcal{CT}_k^{XY}}. \tag{37}$$

Thus, we need to calculate the cycle indices of the three rooted forms of $\mathfrak{a}_k$. From Theorem 3.4.3 and by Theorem 1.5.11 we obtain:

THEOREM 3.5.2. *For the various forms of the species* $\mathfrak{a}_k$ *as in Section 3.2 and the various* $\mathfrak{S}_k$*-species and* $\mathfrak{S}_{k+1}$*-species forms of* $\mathcal{CT}_k$ *as in Section 3.4.1, we have*

$$Z_{\mathfrak{a}_k^Y} = \overline{Z^{\mathfrak{S}_k}_{\mathcal{CT}_k^Y}} = \frac{1}{k!}\sum_{\sigma\in\mathfrak{S}_k} Z^{\mathfrak{S}_k}_{\mathcal{CT}_k^Y}(\sigma) \tag{38a}$$

$$Z_{\mathcal{CT}_k^{XY}} = \overline{Z^{\mathfrak{S}_k}_{\mathcal{CT}_k^{XY}}} = \frac{1}{k!}\sum_{\sigma\in\mathfrak{S}_k} Z^{\mathfrak{S}_k}_{\mathcal{CT}_k^{XY}}(\sigma) \tag{38b}$$

$$Z_{\mathfrak{a}_k^X} = \overline{Z^{\mathfrak{S}_{k+1}}_{\mathcal{CT}_k^{XY}}} = \frac{1}{(k+1)!}\sum_{\sigma\in\mathfrak{S}_{k+1}} Z^{\mathfrak{S}_{k+1}}_{\mathcal{CT}_k^{XY}}(\sigma) \tag{38c}$$

We thus need only calculate the various Γ-cycle indices for the $\mathfrak{S}_k$-species and $\mathfrak{S}_{k+1}$-species forms of $\mathcal{CT}_k^Y$ and $\mathcal{CT}_k^{XY}$ to complete our enumeration of general k-trees.

In Observation 3.3.5, the functional equations for the ordinary species $\mathcal{CT}_k^Y$ and $\mathcal{CT}_k^{XY}$ both include terms of the form $\mathcal{L}_k \circ \mathcal{CT}_k^Y$. The plethysm of ordinary species

does have a generalization to Γ-species, as given in Definition 1.5.7, but it does not correctly describe the manner in which $\mathfrak{S}_k$ acts on linear orders of $\mathcal{CT}_k^{\mathcal{Y}}$-structures in these recursive decompositions. Recall from Section 1.5 that, for two Γ-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 $\mathfrak{S}_k$, however, the actions of σ on the descendant $\mathcal{CT}_k^{\mathcal{Y}}$-structures are *not* independent—they depend on the position of the structure in the linear ordering around the parent black vertex. In particular, if σ acts on some non-root black vertex, then $\rho_i(\sigma)$ acts on the white vertex in the ith place, where in general $\rho_i(\sigma) \neq \sigma$.

Thus, we consider automorphisms of these $\mathfrak{S}_k$-structures directly. First, we consider the component species $X \cdot \mathcal{L}_k\left(\mathcal{CT}_k^{\mathcal{Y}}\right)$.

Lemma 3.5.3. *Let B be a structure of the species $X \cdot \mathcal{L}_k\left(\mathcal{CT}_k^{\mathcal{Y}}\right)$. Let W_i be the $\mathcal{CT}_k^{\mathcal{Y}}$-structure in the ith position in the linear order. Then some $\sigma \in \mathfrak{S}_k$ acts as an automorphism of B if and only if, for each $i \in [k+1]$, we have $\Delta^{-1}(\rho_i(\Delta\sigma))\, W_i \cong W_{\sigma(i)}$.*

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

Consider a structure T of the $\mathfrak{S}_k$-species $\mathcal{CT}_k^{\mathcal{Y}}$ and an element $\sigma \in \mathfrak{S}_k$. As discussed in Section 3.3.3, T is composed of a Y-label and a set of $X \cdot \mathcal{L}_k\left(\mathcal{CT}_k^{\mathcal{Y}}\right)$-structures. The permutation σ acts trivially on Y and $\mathcal{E}$ and acts on each of the component $X \cdot \mathcal{L}_k\left(\mathcal{CT}_k^{\mathcal{Y}}\right)$-structures independently. For each of these component structures, by Lemma 3.5.3, we have that σ is an automorphism if and only if $\Delta\sigma$ carries each $\mathcal{CT}_k^{\mathcal{Y}}$-structure to its $\Delta^{-1}(\rho_i(\Delta\sigma))$-image. Thus, when constructing σ-invariant $X \cdot \mathcal{L}_k\left(\mathcal{CT}_k^{\mathcal{Y}}\right)$-structures, we must construct for each cycle of σ a $\mathcal{CT}_k^{\mathcal{Y}}$-structure which is invariant under the application of *all* the permutations $\Delta^{-1}(\rho_i(\Delta\sigma))$ which will be applied to it along the cycle. For c the chosen cycle of σ, this permutation is $\Delta^{-1}\left(\prod_{i \in c} \rho_i(\Delta\sigma)\right)$, where the product is taken over any chosen linearization of the cyclic order of the terms in the cycle. Once a choice of such a $\mathcal{CT}_k^{\mathcal{Y}}$-structure for each cycle of σ is made, we can simply insert the structures into the $\mathcal{L}_k$-structure to build the desired σ-invariant $X \cdot \mathcal{L}_k\left(\mathcal{CT}_k^{\mathcal{Y}}\right)$-structure. Accordingly:

THEOREM 3.5.4. *The $\mathfrak{S}_k$-cycle index for the species $\mathcal{CT}_k^Y$ is characterized by the recursive functional equation*

$$Z_{\mathcal{CT}_k^Y}^{\mathfrak{S}_k}(\sigma) = p_1[y] \times Z_{\mathcal{E}} \circ \left(p_1[x] \cdot \prod_{c \in C(\sigma)} Z_{\mathcal{CT}_k^Y}^{\mathfrak{S}_k}\left(\Delta^{-1} \prod_{i \in c} \rho_i(\Delta\sigma) \right) \left(p_{|c|}[x], p_{2|c|}[x], \ldots; p_{|c|}[y], p_{2|c|}[y], \ldots \right) \right). \tag{39}$$

where $C(\sigma)$ denotes the set of cycles of σ (as a k-permutation) and the inner product is taken with respect to any choice of linearization of the cyclic order of the elements of c.

The situation for the $\mathfrak{S}_{k+1}$-species $\mathcal{CT}_k^{XY}$ is almost identical. Recall from Section 3.4.1 that $\sigma \in \mathfrak{S}_{k+1}$ acts on a $\mathcal{CT}_k^{XY}$-structure T by applying σ directly to the linear order on the $k+1$ white neighbors of the root black vertex and applying ρ-variants of σ recursively to their descendants. Thus, we once again need only require that, along each cycle of σ, the successive white-vertex structures are pairwise isomorphic under the action of the appropriate $\rho_i(\sigma)$. Thus, we again need only choose for each cycle $c \in C(\sigma)$ a $\mathcal{CT}_k^Y$-structure which is invariant under $\prod_{i \in c} \rho_i(\sigma)$. Accordingly:

THEOREM 3.5.5. *The $\mathfrak{S}_{k+1}$-cycle index for the species $\mathcal{CT}_k^{XY}$ is given by*

$$Z_{\mathcal{CT}_k^{XY}}^{\mathfrak{S}_{k+1}}(\sigma) = p_1[x] \times \prod_{c \in C(\sigma)} Z_{\mathcal{CT}_k^Y}^{\mathfrak{S}_k}\left(\prod_{i \in c} \rho_i[\sigma] \right) \left(p_{|c|}[x], p_{2|c|}[x], \ldots, p_{|c|}[y], p_{2|c|}[y], \ldots \right). \tag{40}$$

under the same conditions as Theorem 3.5.4.

Terms of the form $\prod_{i \in c} \rho_i(\sigma)$ appear in Eqs. (39) and (40). For the simplification of calculations, we note here a two useful results about these products.

First, we observe that certain ρ-maps preserve cycle structure:

Lemma 3.5.6. *Let $\sigma \in \mathfrak{S}_k$ be a permutation of which $i \in [k]$ is a fixed point and let λ be the map sending each permutation in $\mathfrak{S}_k$ to its cycle type as a partition of k. Then $\lambda(\rho_i(\sigma)) = \lambda(\sigma)$.*

PROOF. Suppose $i + a \in [k]$ is in an l-cycle of σ. Then

$$\begin{aligned}
(\rho_i(\sigma))^j(a) &= (\rho_i(\sigma))^{j-1}(\sigma(i+a) - \sigma(i)) \\
&= (\rho_i(\sigma))^{j-2}(\sigma(i + \sigma(i+a) - \sigma(i)) - \sigma(i)) \\
&= (\rho_i(\sigma))^{j-2}\left(\sigma^2(i+a) - \sigma^2(i)\right) \\
&\vdots \\
&= \sigma^j(i+a) - \sigma^j(i)
\end{aligned}$$

But the values of $(\rho_i(\sigma))^j(a) = \sigma^j(i+a) - \sigma^j(i)$ are all distinct for $j \leqslant l$, since $i+a$ is in an l-cycle and i is a fixed point of σ. Furthermore, $(\rho_i(\sigma))^l(a) = \sigma^l(i+a) = i+a$. Thus, a is in an l-cycle of $\rho_i(\sigma)$. This establishes a length-preserving bijection between cycles of $\rho_i(\sigma)$ and cycles of σ, so their cycle types are equal. □

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

Lemma 3.5.7. *Let $\sigma \in \mathfrak{S}_k$ be a permutation with a cycle c. Then $\lambda\left(\prod_{i\in c}\rho_i(\sigma)\right)$ is determined by $\lambda(\sigma)$ and $|c|$.*

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

$$\begin{aligned}
\prod_{i=1}^{|c|}\rho_{c_i}(\sigma) =& \rho_{c_{|c|}}(\sigma)\circ\cdots\circ\rho_{c_2}(\sigma)\circ\rho_{c_1}(\sigma)\\
=& \rho_{c_{|c|}}(\sigma)\circ\cdots\circ\rho_{c_2}(\sigma)\left(a\mapsto\sigma(c_1+a)-\sigma(c_1)\right)\\
=& \rho_{c_{|c|}}(\sigma)\circ\cdots\circ\rho_{c_3}(\sigma)\left(a\mapsto\sigma(c_2+\sigma(c_1+a)-\sigma(c_1))-\sigma(c_2)\right)\\
=& \rho_{c_{|c|}}(\sigma)\circ\cdots\circ\rho_{c_3}(\sigma)\left(a\mapsto\sigma^2(c_1+a)-\sigma^2(c_1)\right)\\
&\vdots\\
=& a\mapsto\sigma^{|c|}(c_1+a)-\sigma^{|c|}(c_1)\\
=& \rho_{c_1}\left(\sigma^{|c|}\right).
\end{aligned}$$

But c_1 is a fixed point of $\sigma^{|c|}$, so by the result of Lemma 3.5.6, this has the same cycle structure as $\sigma^{|c|}$, which in turn is determined by $\lambda(\sigma)$ and $|c|$ as desired. □

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

THEOREM 3.5.8. *$Z^{\mathcal{CT}_k^Y}_{\mathfrak{S}_k}(\sigma)$ and $Z^{\mathcal{CT}_k^{XY}}_{\mathfrak{S}_{k+1}}(\sigma)$ are class functions of σ (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.

3.5.2. k-trees: $\mathfrak{a}_k$. We now have all the pieces in hand to apply Theorem 3.5.1 to compute the cycle index of the species $\mathfrak{a}_k$ of general k-trees. Equation (37) characterizes the cycle index of the generic k-tree species $\mathfrak{a}_k$ in terms of the cycle indices of the rooted species $\mathfrak{a}_k^X$, $\mathfrak{a}_k^Y$, and $\mathcal{CT}_k^{XY}$; Theorem 3.4.3 gives the cycle indices of these three rooted species in terms of the Γ-cycle indices $Z^{\mathfrak{S}_k}_{\mathcal{CT}_k^Y}$, $Z^{\mathfrak{S}_k}_{\mathcal{CT}_k^{XY}}$, and $Z^{\mathfrak{S}_{k+1}}_{\mathcal{CT}_k^{XY}}$;

and, finally, Theorems 3.5.4 and 3.5.5 give these Γ-cycle indices explicitly. By tracing the formulas in Eqs. (39) and (40) back through this sequence of functional relationships, we can conclude:

THEOREM 3.5.9 (Cycle index for the species of k-trees). *For $\mathfrak{a}_k$ the species of general k-trees, $Z_{\mathcal{CT}_k^Y}^{\mathfrak{S}_k}$ as in Eq.* (39)*, and $Z_{\mathcal{CT}_k^{XY}}^{\mathfrak{S}_{k+1}}$ as in Eq.* (40) *we have:*

$$Z_{\mathfrak{a}_k} = \frac{1}{(k+1)!}\sum_{\sigma\in\mathfrak{S}_{k+1}} Z_{\mathcal{CT}_k^{XY}}^{\mathfrak{S}_{k+1}}(\sigma) + \frac{1}{k!}\sum_{\sigma\in\mathfrak{S}_k} Z_{\mathcal{CT}_k^{Y}}^{\mathfrak{S}_k}(\sigma) - \frac{1}{k!}\sum_{\sigma\in\mathfrak{S}_k} Z_{\mathcal{CT}_k^{XY}}^{\mathfrak{S}_k}(\sigma) \tag{41a}$$

$$= \overline{Z_{\mathcal{CT}_k^{XY}}^{\mathfrak{S}_{k+1}}} + \overline{Z_{\mathcal{CT}_k^{Y}}^{\mathfrak{S}_k}} - \overline{Z_{\mathcal{CT}_k^{XY}}^{\mathfrak{S}_k}}. \tag{41b}$$

Equation (41) in fact represents a recursive system of functional equations, since the formulas for the Γ-cycle indices of $\mathcal{CT}_k^Y$ and $\mathcal{CT}_k^{XY}$ 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.

3.6. Unlabeled enumeration and the generating function $\tilde{\mathfrak{a}}_k(x)$

Equation (41) in Theorem 3.5.9 gives a recursive formula for the cycle index of the ordinary species $\mathfrak{a}_k$ of k-trees. The number of unlabeled k-trees with n hedra is historically an open problem, but by application of Theorem 1.2.3 the ordinary generating function counting such structures can be extracted from the cycle index $Z_{\mathfrak{a}_k}$. 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.

Recall from Theorem 1.5.13 that, for a Γ-species F, the ordinary generating function $\tilde{F}_\gamma(x)$ counting unlabeled γ-invariant F-structures is given by

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

and that the ordinary generating function for counting unlabeled F/Γ-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 here. Each of the two-sort cycle indices in this chapter can be converted to one-sort by substituting $y_i = 1$ for all i. For the rest of this section, we will deal directly with these one-sort versions of the cycle indices.

We begin by considering the explicit recursive functional equations in Theorems 3.5.4 and 3.5.5. In each case, by the above, the ordinary generating function is exactly the result of substituting $p_i[x] = x^i$ into the given formula. Thus, we have:

THEOREM 3.6.1. *For $\mathcal{CT}_k^Y$ the $\mathfrak{S}_k$-species of Y-rooted k-coding trees and $\mathcal{CT}_k^{XY}$ the $\mathfrak{S}_{k+1}$-species of edge-rooted k-coding trees, the corresponding Γ-ordinary generating functions are given by*

$$\widetilde{\mathcal{CT}_k^Y}(\sigma)(x) = \exp\Big(\sum_{n\geqslant 1}\frac{x^n}{n}\cdot\prod_{c\in C(\sigma^n)}\widetilde{\mathcal{CT}_k^Y}\Big(\Delta^{-1}\prod_{i\in c}\rho_i(\Delta\sigma^n)\Big)\Big(x^{|c|}\Big)\Big) \tag{42a}$$

and

$$\widetilde{\mathcal{CT}_k^{XY}}(\sigma)(x) = x\cdot\prod_{c\in C(\sigma)}\widetilde{\mathcal{CT}_k^Y}\Big(\prod_{i\in c}\rho_i(\sigma)\Big)\big(x^{|c|}\big). \tag{42b}$$

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

However, as a consequence of Theorem 3.5.8, we can simplify these expressions significantly:

Corollary 3.6.2. *For $\mathcal{CT}_k^Y$ the $\mathfrak{S}_k$-species of Y-rooted k-coding trees and $\mathcal{CT}_k^{XY}$ the $\mathfrak{S}_{k+1}$-species of edge-rooted k-coding trees, the corresponding Γ-ordinary generating functions are given by*

$$\widetilde{\mathcal{CT}_k^Y}(\lambda)(x) = \exp\Big(\sum_{n\geqslant 1}\frac{x^n}{n}\cdot\prod_{i\in\lambda^n}\widetilde{\mathcal{CT}_k^Y}(\lambda^i)(x^i)\Big) \tag{43a}$$

and

$$\widetilde{\mathcal{CT}_k^{XY}}(\lambda)(x) = x\cdot\prod_{i\in\lambda}\widetilde{\mathcal{CT}_k^Y}(\lambda^i)(x^i) \tag{43b}$$

where λ^i denotes the ith 'partition power' of λ — that is, if σ is any permutation of cycle type λ, then λ^i denotes the cycle type of σ^i — and where $f(\lambda)(x)$ denotes the value of $f(\sigma)(x)$ for every σ of cycle type λ.

As in Theorem 3.5.4, 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 3.5.8, we have that $\widetilde{\mathcal{CT}_k^Y}$ and $\widetilde{\mathcal{CT}_k^{XY}}$ are class functions of σ, so we can restrict our computational attention to cycle-distinct permutations.

Moreover, the cycle index of the species $\mathfrak{a}_k$, as seen in Eq. (41), is given simply in terms of quotients of the cycle indices of the two Γ-species $\mathcal{CT}_k^Y$ and $\mathcal{CT}_k^{XY}$. Thus, we also have:

THEOREM 3.6.3. *For $\mathfrak{a}_k$ the species of k-trees and $\widetilde{\mathcal{CT}_k^Y}$ and $\widetilde{\mathcal{CT}_k^{XY}}$ as in Theorem 3.6.1, we have*

(44)

$$\tilde{\mathfrak{a}}_k(x) = \frac{1}{(k+1)!}\sum_{\sigma\in\mathfrak{S}_{k+1}}\widetilde{\mathcal{CT}_k^{XY}}(\sigma)(x) + \frac{1}{k!}\sum_{\sigma\in\mathfrak{S}_k}\widetilde{\mathcal{CT}_k^{Y}}(x)(\sigma) - \frac{1}{k!}\sum_{\sigma\in\mathfrak{S}_k}\widetilde{\mathcal{CT}_k^{XY}}(\sigma)(x).$$

Again, as a consequence of Theorem 3.5.8 by way of Corollary 3.6.2, we can instead write

Corollary 3.6.4. *For $\mathfrak{a}_k$ the species of k-trees and $\widetilde{\mathcal{CT}_k^Y}$ and $\widetilde{\mathcal{CT}_k^{XY}}$ as in Corollary 3.6.2, we have*

(45)

$$\tilde{\mathfrak{a}}_k(x) = \sum_{\lambda\vdash k+1}\frac{1}{z_\lambda}\widetilde{\mathcal{CT}_k^{XY}}(\lambda)(x) + \sum_{\lambda\vdash k}\frac{1}{z_\lambda}\widetilde{\mathcal{CT}_k^{Y}}(\lambda)(x) - \sum_{\lambda\vdash k}\frac{1}{z_\lambda}\widetilde{\mathcal{CT}_k^{XY}}(\lambda\cup\{1\})(x).$$

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. (41). 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 B.2.

3.7. Special-case behavior for small k

Many of the complexities of the preceding analysis apply only for k sufficiently large. We note here some simplifications that are possible when k is small.

3.7.1. Ordinary trees ($k = 1$). When $k = 1$, an $\mathfrak{a}_k$-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 $\mathfrak{S}_k$ are intended to break. The actions of $\mathfrak{S}_2$ act on ordinary trees rooted at a *directed* edge, with the nontrivial element $\tau \in \mathfrak{S}_2$ acting by reversing this orientation. The resulting decomposition from the dissymmetry theorem in Theorem 3.2.1 and the recursive functional equations of Observation 3.3.5 then clearly reduce to the classical dissymmetry analysis of ordinary trees.

3.7.2. 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 $\mathfrak{S}_2$ by reversal of these orientations gives unoriented 2-trees as its orbits. The defined action of $\mathfrak{S}_3$ 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 ρ_i is the trivial map on $\mathfrak{S}_2$ and that

$\rho_i(\sigma) = (1\,2)$ for $\sigma \in \mathfrak{S}_3$ if and only if σ is an odd permutation, both regardless of i. We then have that:

(46a)

$$Z^{\mathfrak{S}_2}_{\mathcal{CT}^Y_2} = p_1[y] \cdot Z_{\mathcal{E}} \circ \left(p_1[x] \cdot \prod_{c \in C(\sigma)} Z^{\mathfrak{S}_2}_{\mathcal{CT}^Y_2}(e) \left(p_{|c|}[x], p_{2|c|}[x], \ldots; p_{|c|}[y], p_{2|c|}[y], \ldots \right) \right)$$

(46b)

$$Z^{\mathfrak{S}_3}_{\mathcal{CT}^{XY}_2} = p_1[x] \cdot \prod_{c \in C(\sigma)} Z^{\mathfrak{S}_2}_{\mathcal{CT}^Y_2}\left(\rho(\sigma)^{|c|}\right) \left(p_{|c|}[x], p_{2|c|}[x], \ldots; p_{|c|}[y], p_{2|c|}[y], \ldots \right).$$

where, by abuse of notation, we let ρ represent any ρ_i. By the previous, the argument $\rho(\sigma)^{|c|}$ in Eq. (46b) is τ if and only if σ is an odd permutation and c is of odd length. This analysis and the resulting formulas for the cycle index $Z_{\mathfrak{a}_2}$ are essentially equivalent to those derived in **[6]**.

APPENDIX A

Computation in species theory

A.1. Cycle indices of compositional inverse species

In Section 2.5, our results included two references to the compositional inverse $\mathcal{CBP}^{\bullet\langle -1\rangle}$ of the species $\mathcal{CBP}^{\bullet}$. Although we have not explored computational methods in depth here, the question of how to compute the cycle index of the compositional inverse of a specified species efficiently is worth some consideration. Several methods are available, including one developed in [**3**, 4.2.19] as part of the proof that arbitrary species have compositional inverses, but our preferred method is one of iterated substitution.

Suppose that Ψ is a species (with known cycle index) of the form $X + \Psi_2 + \Psi_3 + \ldots$ where Ψ_i is the restriction of Ψ to structures on sets of cardinality i and that Φ is the compositional inverse of Ψ. Then $\Psi \circ \Phi = X$ by definition, but by hypothesis

$$X = \Psi \circ \Phi = \Phi + \Psi_2(\Phi) + \Psi_3(\Phi) + \ldots$$

also. Thus

$$\Phi = X - \Psi_2(\Phi) - \Psi_3(\Phi) - \ldots. \tag{47}$$

This recursive equation is the key to our computational method. To compute the cycle index of Φ to degree 2, we begin with the approximation $\Phi \approx X$ and then substitute it into the first two terms of Eq. (47): $\Phi \approx X - \Psi_2(X)$ and thus $Z_\Phi \approx Z_X - Z_{\Psi_2} \circ Z_X$. All terms of degree up to two in this approximation will be correct. To compute the cycle index of Φ to degree 3, we then take this new approximation $\Phi \approx X - \Psi_2(X)$ and substitute it into the first three terms of Eq. (47). This process can be iterated as many times as are needed; to determine all terms of degree up to n correctly, we need only iterate n times. With appropriate optimizations (in particular, truncations), this method can run very quickly on a personal computer to reasonably high degrees; we were able to compute $Z_{\mathcal{CBP}^{\bullet\langle -1\rangle}}$ to degree sixteen in thirteen seconds.

APPENDIX B

Enumerative tables

B.1. Bipartite blocks

With the tools developed in Chapter 2, we can calculate the cycle indices of the species $\mathcal{NBP}$ of nonseparable bipartite graphs to any finite degree we choose using computational methods. This result can then be used to enumerate unlabeled bipartite blocks. We have done so here using Sage 1.7.4 [**23**] and code listed in Appendix C.1. The resulting values appear in Table 1.

TABLE 1. Enumerative data for unlabeled bipartite blocks with n hedra

n	Unlabeled
1	1
2	1
3	0
4	1
5	1
6	5
7	8
8	42
9	146
10	956

B.2. k-trees

With the recursive functional equations for cycle indices of Section 3.5, we can calculate the explicit cycle index for the species $\mathfrak{a}_k$ 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 $k \leqslant 7$ and $n \leqslant 30$ using Sage 1.7.4 [**23**] using code available in Appendix C.2. The resulting values appear in Table 2.

We note that both unlabeled and hedron-labeled enumerations of k-trees stabilize:

THEOREM B.2.1. *For $k \geqslant n + 2$, the numbers of unlabeled and hedron-labeled k-trees are independent of k.*

PROOF. We show that the species $\mathfrak{a}_k$ and $\mathfrak{a}_{k+1}$ 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 2. 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	1
3	2
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	24927175920361855
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	927910207739261
23	6945172081954449
24	52225283886702922
25	394398440097305861
26	2990207055800156659
27	22753619938517594709
28	173727411594289881739
29	1330614569159767263501
30	10221394007530945428347

(D) $k = 4$

n	Unlabeled 4-trees
0	1
1	1
2	1
3	2
4	5
5	15
6	64
7	331
8	2150
9	15817
10	127194
11	1077639
12	9466983
13	85252938
14	782238933
15	7283470324
16	68639621442
17	653492361220
18	6276834750665
19	60759388837299
20	592227182125701
21	5808446697002391
22	57289008242377068
23	567939935463185078
24	5656700148512008902
25	56583199285317631541
26	568236762643725657852
27	5727423267612393252616
28	57924486783495226147615
29	587672090447840337304025
30	5979782184127687211698807

Enumerative data for k-trees with n hedra, continued

(E) $k = 5$

n	Unlabeled 5-trees
0	1
1	1
2	1
3	2
4	5
5	15
6	64
7	342
8	2321
9	18578
10	168287
11	1656209
12	17288336
13	188006362
14	2105867058
15	24108331027
16	280638347609
17	3310098377912
18	39462525169310
19	474697793413215
20	5754095507495584
21	70216415130786725
22	861924378411516159
23	10636562125193377459
24	131890971196221692874
25	1642577274341274449247
26	20538830517384955820622
27	257767439475728146293796
28	3246108646710813383678978
29	410085811895526375400387 47
30	51959949719354740584386 4376

(F) $k = 6$

n	Unlabeled 6-trees
0	1
1	1
2	1
3	2
4	5
5	15
6	64
7	342
8	2344
9	19090
10	179562
11	1878277
12	21365403
13	258965451
14	3294561195
15	43472906719
16	589744428065
17	8171396893523
18	115094557122380
19	1642269376265063
20	23679803216530017
21	344396036645439675
22	5045351124912000756
23	74375422235109338507
24	1102368908826371717478
25	16417712341047912048640
26	245566461812077209025580
27	3687384661929075391318298
28	55566472746158319169779382
29	840092106663809502446963972
30	12739517442131428048314937036

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	1922215
12	22472875
13	284556458
14	3849828695
15	54974808527
16	819865209740
17	12655913153775
18	200748351368185
19	3253193955012557
20	53619437319817482
21	895778170144927928
22	15129118461773051724
23	257812223121779545108
24	4426056869082751747930
25	76463433541541506345648
26	1328088941166844504424628
27	231757966980132120393339479
28	4061035635628648906700292228
29	71423502904686218498140340 57
30	1260349239036999365819345698783

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	1498874117898285574
22	28195965395340358096
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	4019973352
15	59642496465
16	941751344429
17	15724551551655
18	275926445572426
19	5057692869843759
20	96275031338911591
21	1892687812366295682
22	38234411627616084843
23	790120238796588845615
24	16638524087850961727575
25	355878246778832856290372
26	7710423952280397990026132
27	168843592748278228259801752
28	3730285520855433827693340329
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	2056425123910104429
22	43127730369661586804
23	933229734601789336024
24	20749443766669472108394
25	472211306357077710523863
26	10961384502758318928846970
27	258737420965101611169934566
28	6193917223279376307682721853
29	150039339181032274342778699887
30	3670778410024403632885217999313

APPENDIX C

Code listing

Our results in Chapters 2 and 3 provide a framework for enumerating bipartite blocks and general k-trees. However, there is significant work to be done adapting the theory into practical algorithms for computing the actual numbers of such structures. Using the computer algebra system Sage 1.7.4 **[23]**, we have done exactly this. In each case, the script listed may be run with Sage by invoking

```
> sage --python scriptname.py args
```

on a computer with a functioning Sage installation. Alternatively, each code snippet may be executed in the Sage 'notebook' interface starting at the comment "`MATH BEGINS HERE`"; in this case, the final `print...` invocation should be replaced with one specifying the desired parameters.

C.1. Bipartite blocks

The functional Eq. (29) characterizes the cycle index of the species $\mathcal{NBP}$ of bipartite blocks. Python/Sage code to compute the coefficients of the ordinary generating function $\widetilde{\mathcal{NBP}}(x)$ of unlabeled bipartite blocks explicitly follows in listing C.1. Specifically, the generating function may be computed to degree n by invoking

```
> sage --python bpblocks.py n
```

on a computer with a functioning Sage installation.

LISTING C.1. Sage code to compute numbers of bipartite blocks (`bpblocks.py`)

```
1  from sage.all import *
   import sys
3
   # Take arguments from the command line
5  args = sys.argv

7  assert len(args) == 2

9  nval = int(args[1])

11 ##MATH BEGINS HERE
   # Set up a ring of symmetric functions
```

```
p = SymmetricFunctionAlgebra(QQ, basis="power")
x = PowerSeriesRing(QQ, 'x').gen()

# Define basic objects
Zx = p[1]

Z2 = SymmetricGroup(2)
e = Z2.identity()
t = Z2.gen() #The non-identity element

# Utility function for composing a generating function
  into a cycle index
def gf_pleth(f, gf):
    partmapper = lambda part: prod(gf.subs({x:x^i}) for i
      in part)
    return p._apply_module_morphism(f, partmapper)

# Utility function for computing the ogf of unlabeled
  structures of a species with cycle index f
def unlabeled(f):
    partmapper = lambda part: x^(part.size())
    return p._apply_module_morphism(f, partmapper)

# Utility function for computing the egf of labeled
  structures of a species with cycle index f
def labeled(f):
    def partmapper(part):
        if part.length() == part.size():
            return x^(part.length())
        else:
            return 0
    return p._apply_module_morphism(f, partmapper)

# Compute a plethystic inverse of the symmetric function f
  to degree n
def plethystic_inverse(f, n):
        parent = f.parent()

        fstripped = f - f.restrict_degree(1, exact=True)

        improver = lambda F, i: Zx -
          fstripped.restrict_degree(i,
          exact=False).plethysm(F).restrict_degree(i,
          exact=False)

```

```
        finv = Zx

        for i in xrange(2, n+1):
            finv = improver(finv, i)

        return finv

# Utility function to compute the pointing of a cycle index
def pointed(f):
    return Zx*f.derivative_with_respect_to_p1()

# Compute the cycle index of the species E of sets
def Ze(n):
    return balanced_sum(1/part.aut() * prod(p[l] for l in
      part) for i in xrange(1,n+1) for part in
      Partitions(i))

# Compute the cycle index of the species Con which is the
  plethystic inverse of E
def Zcon(n):
    return plethystic_inverse(Ze(n), n)

# Utility function to compute the union of partitions
def union( mu, nu):
    return Partition(sorted(mu.to_list() + nu.to_list(),
      reverse=true))

# Utility function to compute the partwise multiple of a
  partition
def partmult( mu, n ):
    return Partition([part * n for part in mu.to_list()])

# Compute the number of bicolored graphs invariant under a
  permutation of cycle type mu acting on white vertices
  and a permutation of cycle type nu acting on black
  vertices
def efixedbcgraphs( mu, nu ):
    return 2^(sum([gcd(i, j) for i in mu for j in nu]))

# Compute the number of bicolored graphs for which a
  permutation of cycle type mu is a color-reversing
  isomorphism
def tfixedbcgraphs( mu ):
```

```
83      return 2^(len(mu) + sum([integer_ceil(p/2) for p in
          mu]) + sum([gcd(mu[i], mu[j]) for i in range(0,
          len(mu)) for j in range(i+1, len(mu))]))

85  # Compute the Z2-cycle index of the species BC of
      bicolored graphs
    def Zbc( z2elt, n ):
87      assert z2elt in Z2
        if z2elt == Z2.identity():
89          return add([efixedbcgraphs(pair[0], pair[1]) *
              p(union(pair[0],pair[1])) / (pair[0].aut() *
              pair[1].aut()) for i in range(1, n+1) for pair
              in PartitionTuples(i, 2).list()])
        else:
91          return add([tfixedbcgraphs( mu ) * p(partmult(mu,
              2))/partmult(mu, 2).aut() for i in range(1,
              integer_floor(n/2)+1) for mu in
              Partitions(i).list() ])

93  # Compute the Z2-cycle index of the species CBC of
      connected bicolored graphs
    def Zcbc( z2elt, n ):
95      assert z2elt in Z2
        if z2elt == Z2.identity():
97          return Zcon(n).plethysm(Zbc(e,
              n)).restrict_degree(n, exact=False)
        else:
99          scale_part = lambda n: lambda m: m.__class__([i*n
              for i in m])
            pn_pleth = lambda f, n:
              f.map_support(scale_part(n))
101         f = lambda part: prod(pn_pleth(Zbc(t^i, n), i) for
              i in part)
            return p._apply_module_morphism(Zcon(n),
              f).restrict_degree(n, exact=False)
103
    # Compute the cycle index of the species CBP of connected
      bipartite graphs
105 def Zcbp( n ):
        return 1/2 * (Zcbc(e, n) + Zcbc(t, n))
107
    # Compute the ordinary generating function for
      nonseparable bipartite graphs
109 def Znbp_ogf( n ):
        Zcbp_ci = Zcbp(n)
```

```
111     Zcbp_dotinv_ogf =
          unlabeled(plethystic_inverse(pointed(Zcbp_ci), n))
        Zcbp_comp_ogf = gf_pleth(Zcbp_ci,
          Zcbp_dotinv_ogf).add_bigoh(n+1)
113
        Znbp_prime_ogf = gf_pleth(Zcon(n), (x/Zcbp_dotinv_ogf
          - 1).add_bigoh(n+1)).add_bigoh(n) + 1
115
        return Zcbp_comp_ogf + x * Znbp_prime_ogf - x
117
    # Print the result
119 print Znbp_ogf(nval)
```

C.2. *k*-trees

The recursive functional equations in Eqs. (43a), (43b) and (45) characterize the ordinary generating function $\tilde{\mathfrak{a}}_k(x)$ for unlabeled general k-trees. Python/Sage code to compute the coefficients of this generating function explicitly follows in listing C.2. Specifically, the generating function for unlabeled k-trees may be computed to degree n by invoking

```
> sage --python ktrees.py k n
```

on a computer with a functioning Sage installation.

This code uses the class-function optimization of Theorem 3.5.8 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 2b were computed on a modern desktop-class computer in approximately two minutes.

LISTING C.2. Sage code to compute numbers of k-trees (`ktrees.py`)

```
1  from sage.all import *
   import sys
3
   # Take arguments from the command line
5  args = sys.argv

7  assert len(args) == 3

9  kval = int(args[1])
   nval = int(args[2])
11
   ##MATH BEGINS HERE
13 # Set up a ring of formal power series
   psr = PowerSeriesRing(QQ, 'x')
15 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:
        ystretcher = lambda c, part: unlY(Partition(partition_power(part, c)), floor((n-1)/c)).subs({x:x**c})
        descendant_pseries = lambda part: prod(ystretcher(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:
        ystretcher = lambda c: unlY(Partition(partition_power(mu, c)[:-1]), floor((n-1)/c)).subs({x:x**c})
        return (x * prod(ystretcher(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)
```

```
45
    # Print the result
47  print a(kval, nval)
```

Bibliography

1. Lowell W. Beineke and Raymond E. Pippert, *Multidimensional bipartite trees*, Discrete Mathematics **272** (2003), 17–26.
2. L.W. Beineke and R.E. Pippert, *The number of labeled k-dimensional trees*, Journal of Combinatorial Theory **6** (1969), no. 2, 200–205.
3. F. Bergeron, G. Labelle, and P. Leroux, *Combinatorial species and tree-like structures*, Encyclopedia of Mathematics and its Applications, vol. 67, Cambridge University Press, Cambridge, 1998, Translated from the 1994 French original by Margaret Readdy, With a foreword by Gian-Carlo Rota. MR MR1629341 (2000a:05008)
4. Michel Bousquet, *Espèces de structures et applications au dénombrement de cartes et de cactus planaires*, Ph.D. thesis, Université du Québec à Montréal, 1999, Available from LaCIM at `http://lacim.uqam.ca/publications` and as Publications du LaCIM 24.
5. G. W. Ford and G. E. Uhlenbeck, *Combinatorial problems in the theory of graphs. I*, Proc. Nat. Acad. Sci. U. S. A. **42** (1956), 122–128. MR 0078683 (17,1231a)
6. Tom Fowler, Ira Gessel, Gilbert Labelle, and Pierre Leroux, *The specification of 2-trees*, Adv. in Appl. Math. **28** (2002), no. 2, 145–168. MR MR1888841 (2003d:05049)
7. Ira M. Gessel and Gilbert Labelle, *Lagrange inversion for species*, J. Combin. Theory Ser. A **72** (1995), no. 1, 95–117. MR 1354969 (96h:05010)
8. Phil Hanlon, *The enumeration of bipartite graphs*, Discrete Math. **28** (1979), no. 1, 49–57. MR 542935 (81c:05051)
9. F. Harary and E. Palmer, *Graphical enumeration*, Academic Press, New York, 1973.
10. F. Harary and E.M. Palmer, *On acyclic simplicial complexes*, Mathematika **15** (1968), 115–122.
11. F. Harary and R. W. Robinson, *Labeled bipartite blocks*, Canad. J. Math. **31** (1979), no. 1, 60–68. MR 518706 (80b:05036)
12. Frank Harary, *On the number of bi-colored graphs*, Pacific J. Math. **8** (1958), 743–755. MR 0103834 (21 #2598)
13. Frank Harary and Geert Prins, *Enumeration of bicolourable graphs*, Canad. J. Math. **15** (1963), 237–248. MR 0165512 (29 #2794)
14. Anthony Henderson, *Species over a finite field*, J. Algebraic Combin. **21** (2005), no. 2, 147–161. MR MR2142405 (2006c:05017)
15. André Joyal, *Une théorie combinatoire des séries formelles*, Advances in Mathematics **42** (1981), 1–82.
16. Johannes Köbler and Sebastian Kuhnert, *The isomorphism problem for k-trees is complete for logspace*, Proceedings of the 34th International Symposium on Mathematical Foundations of Computer Science 2009 (Berlin, Heidelberg), Springer-Verlag, 2009, pp. 537–548.
17. Saunders Mac Lane, *Categories for the working mathematician*, Graduate texts in mathematics, Springer, 1998.
18. Ji Li, *Counting prime graphs and point-determining graphs using combinatorial theory of species*, Ph.D. thesis, Brandeis University, 2007.
19. J.W. Moon, *The number of labeled k-trees*, Journal of Combinatorial Theory **6** (1969), no. 2, 196–199.
20. E. Palmer, *On the number of labeled* 2*-trees*, Journal of Combinatorial Theory **6** (1969), 206–207.

21. E. Palmer and R. Read, *On the number of plane* 2*-trees*, Journal of the London Mathematical Society **6** (1973), 583–592.
22. Robert W. Robinson, *Enumeration of non-separable graphs*, Journal of Combinatorial Theory **9** (1970), no. 4, 327–356.
23. W. A. Stein et al., *Sage Mathematics Software (Version 1.7.4)*, The Sage Development Team, 2011, `http://www.sagemath.org`.
24. Leopold Travis, *Graphical enumeration: A species-theoretic approach*, Ph.D. thesis, Brandeis University, 1999.