Notes on exponential generating functions and structures.

1. The concept of a structure.

Consider the following counting problems: (1) to find for each n the number of partitions of an n-element set, (2) to find for each n the number of permutations of an n-element set, (3) to find for each n the number of subsets of an n-element set, (4) to find for each n the number of labelled trees on a set of n vertices.

The preceding problems are of a different nature than those we typically solve using ordinary generating functions. The difference is that the things to be counted are tied to a specified n-element set for each n. For instance, if we wish to list all partitions of a 5-element set, we can do it only by introducing a particular 5-element set to work with. Thus we might use the set $\{a, b, c, d, e\}$ and begin listing the partitions $\{\{a, b\}, \{c, d, e\}\}$, $\{\{c\}, \{a, b, d, e\}\}$, and so on. On the other hand if we wish to list all partitions of the *number* 5, there is no underlying set of 5 elements to be introduced; we just list the possibilities: (5), (4, 1) and so on.

What is common to the problems listed at the outset is that in each of them, we begin for each n with a set of n elements, and then we are asked to count all possible ways of imposing a certain *structure* upon that given set. Thus for a partition, we are to structure the set by arranging its elements into blocks. For a permutation, we are to arrange them in some order. For a subset, we are to arrange them into those chosen to belong to the subset, and those not. For a tree, we are to arrange the elements into the vertices of a tree.

The purpose of exponential generating functions is to solve counting problems like these, in which some kind of structure is specified and we are asked to find for each n the number of ways to arrange the elements of an n-element set into such a structure. We define the *exponential generating function* associated with a counting problem for structures to be

$$F(x) = \sum_n f(n)\frac{x^n}{n!},$$

where $f(n)$ is the number of structures on an n-element set. For the moment, we will have to accept this on faith as a good definition. As we proceed further, however, this definition will be justified by the fact that it leads to addition and multiplication principles that are appropriate to structure-counting problems. As we shall see, it also leads to important new counting principles that go beyond what we can do with addition and multiplication.

2. Trivial structures

Before it is possible to work out any significant examples of exponential generating functions it will be necessary to build a small repetoire of generating functions for some seemingly stupid structures. We call them *trivial* structures because the structures themselves have little content. Nevertheless we will find that their generating functions are not in any sense "trivial," but may be interesting functions. Furthermore these structures are not at all "trivial" in their usefulness, for every structure we consider will have these as building blocks at some level.

We begin with *the trivial structure*, which is the structure of "being a set." What we mean is that to impose the trivial structure on a given set is to give the elements only that structure which they already have: the structure of a set. The point is, there is just one such structure on every set. Thus the number of trivial "being a set" structures on an n-element set is $f(n) = 1$ for every n. The exponential generating function for this trivial structure is therefore

$$F(x) = \sum_{n=0}^{\infty} \frac{x^n}{n!} = e^x.$$

Note that the generating function for this most trivial structure is not at all a trivial function.

More generally, we can consider trivial structures that express restrictions on the size of a set. The simplest such restriction is the structure of "being a 1-element set." To impose this structure on a 1-element set is to give it only the structure it already has—but to impose it on anything but a one element set is impossible. In short, there is by definition exactly one structure of "1-element set" on every 1-element set, and none on any other set. The number of trivial "1-element set" structures on an n-element set is therefore $f(n) = 1$ if $n = 1$, and $f(n) = 0$ otherwise. The corresponding generating function is

$$F(x) = \sum_n f(n)\frac{x^n}{n!} = x.$$

In a similar way, corresponding to any restriction we might care to place on the size of a set we can define a trivial structure reflecting that restriction. By definition there is to be one such structure on a set if it obeys the restricition, and none if it doesn't. The number $f(n)$ of our trivial structures on an n-element set is thus, by definition, the *indicator function* that is 1 for good values of n and 0 for bad values. The exponential generating function $F(x) = \sum_n f(n)x^n/n!$ for our trivial structure is then simply the sum of $x^n/n!$ taken over all allowed values of n. Fortunately, in many cases this is simple to express in closed form, as in the two examples we just did.

Here are some examples of trivial structures. Although the first four examples below are quite trivial indeed, they are also the ones that are most often useful.

(1) The trivial structure of "set": $F(x) = e^x$.
(2) The trivial "1-element set" structure: $F(x) = x$. k-element $F_k(x) = \frac{x^k}{k!}$
(3) The trivial "empty set" strucutre: $F(x) = 1$.
(4) The trivial "non-empty set" structure: $F(x) = e^x - 1$.
(5) The trivial "even-size set" structure: $F(x) = \cosh(x) = (e^x + e^{-x})/2$.
(6) The trivial "odd-size set" structure: $F(x) = \sinh(x) = (e^x - e^{-x})/2$.

As an exercise to be sure you understand this so far, I suggest you pause here and work out in each of the above examples why the generating function $F(x)$ is what I said it is.

3. Addition and multiplication principles

To count anything interesting using exponential generating functions, we need to know what it means to add and multiply them. To state the relevant principles, let's suppose that we have given names to some kinds of structures, say f-structures, g-structures, and h-structures. We let $f(n)$ be the number of f-structures on an n-element set, so the exponential generating function counting f-structures is $F(x) = \sum f(n)x^n/n!$. Similarly for g and h.

Addition principle for exponential generating functions: Suppose that the set of f-structures on each set is the disjoint union of the set of g-structures and the set of h-structures. Then

$$F(x) = G(x) + H(x).$$

The addition principle is rather obvious and is no different from the addition principles for other types of enumeration, such as straight counting, or ordinary generating functions. As an example, we can consider f = "trivial" structure, g = "empty" structure, h = "non-empty" structure. Every set is either empty or non-empty, and these possibilities are mutually exclusive, so we have $F(x) = G(x) + H(x)$. In this case, we know that $F(x) = e^x$ and $G(x) = 1$, which yields the formula $H(x) = e^x - 1$ for the exponential generating function of the "non-empty set" structure.

The power of exponential generating functions begins to appear when we consider the multiplication principle. We begin by defining a notion of product of two structures.

Definition: Let g and h denote two types of structures on finite sets. A *$g \times h$ structure* on a set A consists of

(i) an ordered partition of A into disjoint subsets $A = A_1 \cup A_2$,
(ii) a g-structure on A_1, and
(iii) an h-structure on A_2,

where the structures in (ii) and (iii) are chosen independently.

Multiplication principle for exponential generating functions: If $G(x)$ and $H(x)$ are the exponential generating functions for g-structures and h-structures, respectively, then the exponential generating function for $g \times h$ structures is

$$F(x) = G(x)H(x).$$

There is a natural generalization of this principle to the product of three or more generating functions. Namely, a $g_1 \times g_2 \times \cdots \times g_r$ structure on A consists of an ordered partition of A into r disjoint subsets $A = A_1 \cup A_2 \cup \cdots \cup A_r$, and independently chosen g_i structures on each subset A_i. With this definition, to give a $g_1 \times g_2 \times \cdots \times g_r$ structure is equivalent to giving a $g_1 \times (g_2 \times \cdots \times g_r)$ structure, and it then follows by induction on r that the generating function for $g_1 \times g_2 \times \cdots \times g_r$ structures is

$$F(x) = G_1(x)G_2(x)\cdots G_r(x).$$

4. Examples

(1) Let us find the exponential generating function for the number of subsets of an n-element set. To choose a subset of A is equivalent to choosing an ordered partition of A into $A_1 =$ the subset, and $A_2 =$ its complement. No further structure is imposed on A_1 or A_2, which is to say that we have the *trivial* structure on each of them. We apply the multiplication principle with $G(x)$ and $H(x)$ both equal to the trivial structure generating function e^x, obtaining the generating function for subsets as

$$F(x) = e^x e^x = e^{2x} = \sum_n \frac{(2x)^n}{n!} = \sum_n 2^n \frac{x^n}{n!}.$$

Of course this accords with our prior knowledge that an n-element set has 2^n subsets.

(2) Modifying the previous example a bit, let us find the generating function counting *non-empty* subsets of an n-element set. We still have trivial structure on A_2, but we are now imposing the structure of non-empty set on A_1. Thus we get

$$F(x) = (e^x - 1)e^x = e^{2x} - e^x = \sum_n (2^n - 1)\frac{x^n}{n!}.$$

Again we get the expected answer, since we already knew that there are $2^n - 1$ non-empty subsets of an n-element set.

(3) Let us fix a number k and count functions from an n-element set to $\{1, \ldots, k\}$. As a structure on the n-element set A, to give a function $A \to \{1, \ldots, k\}$ is equivalent to giving the ordered partition $A = A_1 \cup \cdots \cup A_k$ in which A_i is the set of elements of A that are mapped by the function to i. In other words, our structure is equivalent to a product of k trivial structures. Therefore we should multiply together k trivial-structure generating functions to get

$$F(x) = (e^x)^k = e^{kx} = \sum_n k^n \frac{x^n}{n!}$$

as the generating function counting functions $A \to \{1, \ldots, k\}$ on an n-element set A. Once again this accords with our prior knowledge that there are k^n such functions.

$\{1, \ldots, n\} \to \{1, \ldots, k\} \cong \{1, \ldots, n\}^k$

ordered partitions

(4) The previous example might seem rather elementary, but consider this slight variation: we will count *surjective* functions from an n-element set A to $\{1,\dots,k\}$. The only difference from the previous example is that now each block A_i in the ordered partition that describes the function must be non-empty. In other words our structure is equivalent to the product of k trivial "non-empty set" structures. We can solve this problem as easily as the last one, replacing e^x for the trivial structure with e^x-1 for the "non-empty set"structure. This yields the generating function

$$F(x)=(e^x-1)^k$$

for surjective functions $A\to\{1,\dots,k\}$ on an n-element set A.

From our study of distribution problems, we know that the number of partitions of an n-element set A into k blocks is (by definition) the Stirling number $S(n,k)$. The number of ordered partitions into k blocks, or equivalently of surjections $A\to\{1,\dots k\}$, is $k!S(n,k)$. Therefore we have the identity

$$(e^x-1)^k=\sum_n k!S(n,k)\frac{x^n}{n!},$$

or

$$\frac{(e^x-1)^k}{k!}=\sum_n S(n,k)\frac{x^n}{n!}.$$

So we have come up with an exponential generating function for the Stirling numbers $S(n,k)$ as n varies, using only extremely simple combinatorial considerations. By contrast an ordinary generating function for these same numbers can only be arrived at through a fairly subtle bijection.

(5) In the last part of example (4) we have found the exponential generating function counting unordered partitions of an n-element set into k blocks, since the number of these is $S(n,k)$. Summing over all values of k we can count *all* partitions of an n-element set. Their exponential generating function is therefore

$$\sum_n B_n\frac{x^n}{n!}=\sum_k\frac{(e^x-1)^k}{k!}=e^{e^x-1}$$

infinite sum principle

$B(x)=e^{e^x-1}$

$B'(x)=B(x)\cdot e^x$ CDA

where the *Bell number* B_n is defined to be the number of all partitions of an n-element set. Here we can take note of a curious fact: the Bell number generating function we just found turns out to have the form $G(H(x))$ with $G(x)=e^x$ the generating function for the trivial structure and $H(x)=e^x-1$ the generating function for the trivial non-empty structure. We shall explain this later on.

(6) We now count permutations of an n-element set. In this example we will identify permuations of A with linear orderings of A, rather than with bijections from A to itself. We will see later on what happens if we take the latter point of view instead. If $n=0$, so the set is empty, we shall agree that there is a unique empty permutation. Otherwise, for a nonempty set, we can choose a permutation recursively as follows: first decide the first element of the permutation, then decide how to permute the remaining elements. Note that this is correct for $n=1$ because of our convention that the empty set has one permutation. We can describe the choices involved nicely in terms of product structures and trivial structures. To choose a first element and a permutation of the remaining elements is to choose a product structure of the form (one-element set)$\times$(permutation). More precisely, such a product structure consists of an ordered partition $A=A_1\cup A_2$ in which A_1 is required to consist of a single element, together with a permutation of the elements of A_2. Our chosen first element becomes the element of A_1 here.

ordered

$B'(x)=e^{e^x-1}\cdot e^x=B(x)\cdot e^x$

$P = 1 \mid 1 \times P \qquad P \leftarrow \varepsilon \mid a \,\text{ш}\, P$

↑ shuffle product

If $F(x)$ stands for the exponential generating function counting permutations we apply the addition and multiplication principles to get the identity

$$F(x) = 1 + xF(x).$$

The first term here is the trivial "empty set" structure counting the empty permutation. The term $xF(x)$ counts permutations of non-empty sets by the multiplication principle, the factor x being the generating function for the trivial "one-element set" structure. Solving for $F(x)$, we have

$$F(x) = 1/(1-x) = \sum_n x^n = \sum_n n! \frac{x^n}{n!},$$

in agreement with our prior knowledge that there are $n!$ permutations of an n-element set. Note that this computation has nothing to do with the "repetition principle" for ordinary generating functions. We are dealing here with $1/(1-x)$ as an exponential generating function, and it doesn't count repetitions of anything.

(7) We have now solved all but one of the counting problems mentioned at the beginning of these notes. The problem not yet solved is that of counting trees on an n-element set of vertices. There are many possible versions of this problem, some of which we will consider in detail later on. For now, we solve only one version of the problem. We will count binary trees whose vertices, including the root, are labelled by the given n-element set. We shall agree that the empty tree counts as a binary tree.

To choose a binary tree on a non-empty vertex set A, we partition A into three parts: the root A_1, the left subtree A_2, and the right subtree A_3. Since there is only one root vertex, the structure on A_1 is the trivial 1-element set structure. The other parts A_2 and A_3 are again given the structure of binary tree. Note that our agreement that the empty tree counts as a tree allows for the possibility that A_2 and/or A_3 are empty. Let us denote the exponential generating function for labelled binary trees by $T(x)$. Then the description we have just given leads to the identity

$$T(x) = 1 + xT(x)^2.$$

Solving for $T(x)$, we get

$$T(x) = \frac{1 - \sqrt{1-4x}}{2x} = \sum_n C_n x^n = \sum_n n! C_n \frac{x^n}{n!},$$

Where C_n is the n-th Catalan number.

At first it might appear that this example is essentially the same as our earlier determination of the number of binary trees on n vertices, using ordinary generating funtions. In fact, however, there is a profound difference. What we counted here was not abstract unlabelled binary trees, but labelled binary tree structures on a given set of vertices. Indeed, we do *not* find that the number of them is C_n, but rather $n!C_n$, since $T(x)$ is an exponential generating function. The confusion results from the accidental fact that the exponential generating function for $n!C_n$ and the ordinary generating function for C_n are the same. This accident occurred because binary trees have an inherent order, so there are exactly $n!$ ways to label an unlabelled binary tree on n verticies. Later, we will use exponential generating functions to count unordered trees, and then this accidental phenomenon will not occur.

5. The functional composition principle

We are now going to discover the real power of exponential generating functions. This power comes from the fact that if $G(x)$ and $H(x)$ are exponential generating functions, then the functional composition $G(H(x))$ has a combinatorial interpretation.

In order for $G(H(x))$ to make sense as a formal power series, $H(x)$ must satisfy $H(0) = 0$, that is, the constant term of the generating function $H(x)$ must be zero. Combinatorially, this means that $H(x)$ must count structures that only exist on non-empty sets, since $h(0)$, the number of h-structures on the empty set, is required to be zero.

Definition: Let g and h be types of structures, and assume that there are no h-structures on the empty set. A *composite $g \circ h$ structure* on a set A consists of

(i) a partition of A into any number of blocks,
(ii) independently chosen h-structures on each block of the partition, and
(iii) a g-structure on the set of blocks.

Here the structure in (iii) is also chosen independently from the structures in (ii).

Composition principle: If the exponential generating functions for g-structures and h-structures are $G(x)$ and $H(x)$, respectively, where $H(0) = 0$, then the exponential generating function for composite $g \circ h$ structures is given by

$$F(x) = G(H(x)).$$

There is an important point to be made about the way in which the partition in a composite structure is chosen. A priori, the partition should be chosen freely, without any restriction on its number of blocks or on their sizes. However, we can always impose implicit restrictions on these by using suitable h- and g-structures in parts (ii) and (iii) of the definition of composite structure. In particular, we can often impose useful restrictions by taking trivial structures for h or g.

A second point relates to the requirement that $H(0) = 0$. Notice that in forming a composite structure, you choose h-structures on the blocks, which are non-empty by definition. In the definition of composite structure it therefore makes no difference how many h-structures there are on the empty set. If we want to use for h a type of structure which does not have $H(0) = 0$, we can always compensate by subtracting the constant term of $H(x)$ from $H(x)$, modifying it to obtain a new series $\widehat{H}(x)$ with $\widehat{H}(0) = 0$. This should only be done with care, however. In most cases, when we have analyzed the problem correctly, the $H(x)$ that goes into an application of the composition principle will naturally have $H(0) = 0$. If it does not, it may be a sign of error in our logic.

6. Composition principle examples

(1) Consider again the Bell number generating function e^{e^x-1} which counts all partitions of an n-element set. Before, we had to get this by summing the generating function for Stirling numbers $S(n, k)$ over all values of k. Using the composition principle, we can get the result directly. The structure of a partition of A is merely a composite structure in which the structures on each block and on the set of blocks are all trivial. However, note that the structure taken on each block is required to be one which only exists on non-empty sets. Therefore we should take g to be the trivial structure and h to be the "non-empty set" structure. This gives $G(x) = e^x$, $H(x) = e^x - 1$ and $F(x) = G(H(x)) = e^{e^x-1}$.

(2) The previous example admits a multitude of variations. For instance, we can count partitions of an n-element set into blocks with more than one element by replacing $H(x)$ with the generating function for the trivial structure of "set of more than one element," which is $e^x - x - 1$. This gives the generating function

$$F(x) = e^{e^x-1-x}$$

≥ 2 elements

In a similar vein, we see that the generating function for partitions with only odd-size blocks is

$$F(x) = e^{\sinh(x)},$$

while the generating function for partitions with only even-size blocks is

$$F(x) = e^{\cosh(x)-1}.$$

total preorders

(3) Suppose you are asked to rate your preferences among n different foods, or sports, or whatever. You may prefer some to others, but there might also be ones you like equally well. Thus your ranking may not be a total ordering of all the items, but rather a partition of the items into "indifference classes," with a linear ordering on the set of classes. Such a structure on the set of items is called a *preference ordering*. We can regard a preference ordering as a composite structure consisting of a partition, a linear ordering on the set of blocks, and a trivial non-empty set structure on each block. We have already found the generating function $G(x) = 1/(1-x)$ for linear orderings. Applying the composition principle with $H(x) = e^x - 1$, we get the generating function for preference orderings

$$F(x) = \frac{1}{1-(e^x-1)} = \frac{1}{2-e^x}.$$

Since this is a new concept we have not encountered before, it is interesting to evaluate the first few terms of the generating function. This can be done by substituting for e^x the first few terms of its Taylor series and using long division to get

$$\frac{1}{2-e^x} = \frac{1}{1-x-x^2/2-x^3/6-\cdots} = 1 + x + 3x^2/2 + 13x^3/6 + \cdots.$$

Thus there is 1 preference order on the empty set, 1 preference order on a set of one item, 3 possible preference orders on a set of 2 items, and 13 on a set of 3 items. For two items, the three possibilities are $(\{a,b\})$, $(\{a\},\{b\})$, and $(\{b\},\{a\})$. For three items, you may wish to pause and list the 13 possibilities yourself.

(4) Here is a an approach to counting permutations different than the one we used earlier. Before, we identified permutations with linear orderings. This time, we identify permutations of a set A with bijections from A to A, and make use of the decomposition of a permutation as a product of disjoint cycles. We define a *cyclic ordering* of an n-element set to be a permutation of its elements consisting of a single cycle of length n. Note that there are $(n-1)!$ cyclic orderings of an n-element set. We shall agree that there is no cyclic ordering of the empty set. Let $L(x)$ be the exponential generating function for cyclic orderings. Since we know how many cyclic orderings there are, we can write this out explicitly as

$$L(x) = \sum_{n=1}^{\infty} (n-1)! \frac{x^n}{n!} = \sum_{n=1}^{\infty} x^n/n.$$

Next consider any permutation of $A = \{1,\ldots,n\}$ expressed in cycle notation. To choose such a permutation we can first choose a partition of A, then arrange each block into a cycle. On the set of cycles there is no further structure—that is, we have a trivial structure. Applying the composition principle, with $G(x) = e^x$ and $H(x) = L(x)$, we find that the generating function for permutations is

$$F(x) = e^{L(x)}.$$

However, we already know that the permutation generating function is equal to $1/(1-x)$, so we get the identity

$$e^{L(x)} = 1/(1-x),$$

and hence

$$L(x) = \log 1/(1-x) = -\log(1-x).$$

In this case, we already knew the answer to the counting problem, and we have used it to derive an identity. Namely, we have found the power series expansion

$$-\log(1-x) = \sum_n x^n/n = x + \frac{x^2}{2} + \frac{x^3}{3} + \cdots$$

by purely combinatorial means, without using calculus. Of course, this is a formal power series expansion, and tells us nothing about the domain of convergence on the right hand side.

7. The derivative principle

Derivative principle: Suppose an f-structure on a set A is equivalent to an h-structure on the set $A \cup \{*\}$ formed by augmenting A with an extra element. Then

$$F(x) = H'(x).$$

Here are some examples:

(1) Suppose we linearly order the elements of A plus an extra element $*$. The position of $*$ in the ordering cuts A into two subsets, so such a structure is equivalent to partitioning A into blocks A_1 and A_2, and putting a linear order on each block. Writing $F(x)$ for the generating function for linear orderings, we obtain the differential equation

$$F'(x) = F(x)^2.$$

Since $F(0) = 1$, we can solve this to get the identity $F(x) = 1/(1-x)$ in yet another way.

(2) We will count *alternating permutations* of $\{1, \ldots, n\}$ for every n. By this we mean a permutation which has the form

$$a_1 < a_2 > a_3 < a_4 > \cdots < a_{n-1} > a_n$$

in one-row notation. Note that we can only have the above pattern if n is odd. In particular, we agree that there is no empty alternating permutation, and that the unique permutation of $\{1\}$ is alternating. Let $F(x)$ be the exponential generating function counting alternating permutations. By listing them, you can verify that there are 2 alternating permutations of $\{1, 2, 3\}$ (namely 132 and 231) and 16 of $\{1, 2, 3, 4, 5\}$, so the first few terms of $F(x)$ are

$$F(x) = x + 2x^3/6 + 16x^5/120 + \cdots .$$

Now consider the problem of choosing an alternating permutation on numbers $\{1, \ldots, n+1\}$, which we think of as $A = [n] = \{1, \ldots, n\}$ augmented with the extra element $n+1$. Of course alternating permuations on $A \cup \{n+1\}$ only exist when n is even. By the derivative principle, the generating function for alternating permutations on $A \cup \{n+1\}$ is $F'(x)$. Note that since $F(x)$ contains only odd powers of x, it follows that $F'(x)$ contains only even powers of x, as it should.

When $n > 0$, to choose an alternating permuation on $A \cup \{n+1\}$, you must put some of the numbers $1, \ldots, n$ before $n+1$ and the remaining numbers after $n+1$. Since an alternating permutation begins with an increase and ends with a decrease, $n+1$ can't be at either end. Thus the portions before and after $n+1$ must both be non-empty. In the portion before $n+1$ the last step must be a decrease, since the step to $n+1$ is automatically an increase. Thus the first portion is an alternating permutation. By similar reasoning, so is the second portion. Thus the set $1, \ldots, n$ receives a product structure: it is partitioned into two parts, each arranged into an alternating permutation. The correspondence between such structures on $1, \ldots, n$ and alternating permutatons on $1, \ldots, n+1$ is clearly bijective. By the multiplication principle, the generating function for such structures is $F(x)^2$, and adding 1 for the case $n = 0$ we arrive at the differential equation

$$F'(x) = F(x)^2 + 1.$$

Together with the initial value $F(0) = 0$, this equation determines $F(x)$. The solution is

$$F(x) = \tan x.$$

Thus we have found a combinatorial interpretation of the coefficients of the power series for $\tan x$—it is the exponential generating function for alternating permutations.

By similar reasoning you can show that $\sec x$ is the exponential generating function for even alternating permutations (which start and end with an increasing step, and include the empty permutation).

8. Proof of the principles

In this section we will prove the counting principles for exponential generating functions. The addition principle is obvious, so we begin with the multiplication principle.

Multiplication principle: Let $f(n)$ be the number of $g \times h$ structures on an n-element set A. To choose a $g \times h$-structure is to choose a partition of A into A_1 and A_2, with g-structure on A_1 and h-structure on A_2. If $|A_1| = k$, then there are $\binom{n}{k}$ choices of the partition, $g(k)$ choices of g-structure, and $h(n-k)$ choices of h-structure. Summing over all k we have

$$f(n) = \sum_{k=0}^{n} \binom{n}{k} g(k)h(n-k) = \sum_{k=0}^{n} n! \frac{g(k)}{k!} \frac{h(n-k)}{(n-k)!}.$$

Hence

$$f(n)/n! = \sum_{k=0}^{n} \frac{g(k)}{k!} \frac{h(n-k)}{(n-k)!}.$$

This last sum is precisely the coefficient of x^n in $G(x)H(x)$, showing that $F(x) = G(x)H(x)$.

Composition principle: We will use the addition and multiplication principles. As part of a $g \circ h$ structure on A we have a partition with some number of blocks, say k. We will find the generating function for $g \circ h$ structures with exactly k blocks, then sum over all k.

Fixing the number of blocks to be k, let us consider the structure on A consisting of an *ordered* partition $A = A_1 \cup \cdots \cup A_k$, together with an h-structure on each block A_i and a g-structure on the set of blocks. There are always $g(k)$ ways to choose the g structure, and the remaining part of the structure is just the product of k h-structures. Hence the generating function for such ordered composite structures with exactly k blocks is

$$g(k)H(x)^k.$$

Note that we have used here the hypothesis that $H(0) = 0$, that is, that there are no h-structures on the empty set. Without this hypothesis, some of the parts A_i in in the product of h-structures might be empty, so that we would not necessarily be counting structures with exactly k blocks.

Now, there are $k!$ ordered structures as above for each composite $g \circ h$ structure with k blocks, so the generating function for the latter is

$$\frac{g(k)}{k!} H(x)^k.$$

Summing over all k we obtain the generating function for composite structures

$$F(x) = \sum_{k} g(k) \frac{H(x)^k}{k!} = G(H(x)).$$

Derivative principle: The derivative of $x^n/n!$ is $x^{n-1}/(n-1)!$. Therefore if $F(x) = \sum f(n)x^n/n!$ then the coefficient of $x^n/n!$ in $F'(x)$ is $f(n+1)$, the number of f-structures on an n-element set A augmented by an extra element.

Notes on exponential generating functions (continued).

9. The exponential generating function for rooted labelled trees

In this section we consider the problem of enumerating unordered rooted trees on a set of n labelled vertices. This is a typical structure enumeration problem which we can attack using exponential generating functions. We have already seen how to enumerate binary trees using the product principle. For unordered trees we typically need the composition principle.

Let $t(n)$ denote the number of unordered rooted labeled trees on an n-element set, and let

$$T(x) = \sum_{n=0}^{\infty} t(n) \frac{x^n}{n!}$$

be the corresponding exponential generating function. It will turn out that things work best if we agree that $t(0) = 0$, that is, we do not count the empty tree as a rooted tree.

We begin as we did for binary trees, by viewing a tree as a product structure, in which one vertex is chosen to be the root, and the rest are arranged into a collection of subtrees whose roots are the children of the main root. In other words, the vertices other than the one chosen as the root are given the structure of a "forest" of rooted trees. Next we observe that a forest is really a composite structure, consisting of a partition of the vertices, with a structure of rooted tree on the vertices in each block, and a trivial structure on the set of blocks.

By the composition principle, the exponential generating function enumerating forests is equal to

$$e^{T(x)}.$$

Note that our agreement not to count the empty tree is convenient here, since it makes $T(0) = 0$, as we require when applying the composition principle. Now the product structure describing a root and a forest is (one-element set)$\times$(forest), so by the product principle we have

$$T(x) = xe^{T(x)}.$$

This identity determines $T(x)$. To actually compute any fixed number of terms, one can use the method of successive approximation. Begin with

$$T(x) = x + \cdots,$$

which is correct through the order x term. Substituting this into the right hand side in the above identity, we will get $e^{T(x)}$ correct through the order x term, and hence we will get $xe^{T(x)}$ correct though the order x^2 term. In this way we find

$$T(x) = xe^{(x+\cdots)} = x(1 + x + \cdots) = x + x^2 + \cdots.$$

Repeating this, we get

$$\begin{aligned} T(x) &= xe^{(x+x^2+\cdots)} = xe^x e^{x^2} \cdots \\ &= x(1 + x + x^2/2 + \cdots)(1 + x^2 + \cdots) \\ &= x(1 + x + 3x^2/2 + \cdots) \\ &= x + x^2 + 3x^3/2 + \cdots. \end{aligned}$$

This can be continued indefinitely, the next step giving

$$T(x) = x + 2x^2/2! + 9x^3/3! + 64x^4/4! + \cdots.$$

This agrees with what we knew from Cayley's tree enumerator or the matrix-tree theorem to be the answer, namely $t(n) = n^{n-1}$. Note however that the exponential generating function approach

allows us to arrive at the identity $T(x) = xe^{T(x)}$ in a direct fashion, whereas to get the Cayley or matrix-tree generating functions, we first needed a kind of amazing guess to discover the answer, and then a tricky argument to prove it.

10. Lagrange inversion

From the computation of $T(x)$ above we might readily guess the formula $t(n) = n^{n-1}$ if we hadn't known it before, but it is still not apparent why the formal power series

$$T(x) = \sum_{n=1}^{\infty} n^{n-1} \frac{x^n}{n!}$$

should be the solution of the equation

$$T(x) = xe^{T(x)}.$$

Here I will briefly discuss one way in which this result can be obtained. We can rewrite the identity for $T(x)$ as

$$T(x)e^{-T(x)} = x,$$

which says that $T(x)$ is the *inverse function* of xe^{-x}, in the same way that $\arcsin x$ is the inverse function of $\sin x$, or e^x is the inverse function of $\log x$. There is a classical formula of Lagrange to find the coefficients of the Taylor series of an inverse function. Since we are concerned here with formal series, we only allow series with zero constant term, as these are the only formal series for which it makes sense to speak of a formal inverse function. Then the *inversion formula of Lagrange* can be written as follows.

Theorem 1. *Let $xG(x)$ be the functional composition inverse of $xF(x)$. Then*

$$[x^n]G(x) = [x^n]\frac{F(x)^{-n-1}}{n+1},$$

where the symbol $[x^n]$ denotes the coefficient of x^n in the expression that follows it.

Note that the formula gives the x^n coefficient of $G(x)$ as the x^n coefficient of another series which *depends on* n. It does not give a closed form for $G(x)$, which would be impossible in general.

At this point, we will not discuss the proof of Lagrange's formula, but take it for granted and apply it in our situation. Since $T(x)$ is the functional composition inverse of xe^{-x}, we take $F(x) = e^{-x}$. Then $G(x) = T(x)/x$, so the coefficient of x^n in $G(x)$ is actually the coefficient of x^{n+1} in $T(x)$, which is $t(n+1)/(n+1)!$. According to the formula, this is given by

$$t(n+1)/(n+1)! = [x^n]\frac{e^{(n+1)x}}{n+1} = \frac{(n+1)^n}{n!(n+1)}.$$

Hence

$$t(n+1) = (n+1)^n,$$

or, replacing n with $n-1$,

$$t(n) = n^{n-1}.$$

11. Variations on tree enumeration

We can use the method of the previous section to count rooted labeled trees with various kinds of additional structure on the children of each vertex. We will always view a tree as a product structure (one-element root)×(forest), and a forest as a composite structure. In general the outer structure on the trees in the forest may be non-trivial, depending on what type of trees we want to count.

Example: Unordered rooted labeled binary trees. These are unordered trees in which every vertex has at most two children. Thus each forest is to be a forest of at most two trees. The generating function for the trivial structure of "set with at most two elements" is $1 + x + x^2/2$, so

$$U(x) = x(1 + U(x) + U(x)^2/2),$$

where $U(x)$ is the generating function for unordered rooted labeled binary trees. Note that this is a quadratic equation which can be solved exactly for $U(x)$.

Example: Unordered rooted labeled strictly binary trees. This means each vertex has exactly two children or none. For the generating function we get

$$V(x) = x(1 + V(x)^2/2).$$

Example (just to show the lengths to which you can take this method): Unordered rooted labeled trees in which every vertex either has only leaves or no leaves as children. As before, we analyze such a tree as a product structure consisting of a root and a forest. The forest structure for this one is a bit tricky. Either the forest is all leaves, that is, it is a forest of one-vertex trees, or else it is a forest of trees of our same type again, all of which have at least two vertices. Let $Z(x)$ be the exponential generating function enumerating our trees. By definition we don't count the empty tree, and there is one tree on a one-element set, contributing a term x to $Z(x)$. The remaining terms of $Z(x)$ enumerate the trees with more than one element, so their generating function is $Z(x) - x$. By the addition principle and the composition principle, the generating function enumerating forests of the type we want is $e^x + e^{Z(x)-x}$. Here the term e^x enumerates the forests consisting of one-vertex trees (a forest of one-vertex trees is just a trivial structure) and the other term enumerates the forests of trees with more than one vertex. The identity giving $Z(x)$ is then

$$Z(x) = x(e^x + e^{Z(x)-x}).$$

12. Trees, permutations, and functional digraphs

Let A be a finite set and $f\colon A \to A$ be any function from A to itself—not necessarily a permutation. We can form a directed graph whose vertices are the elements of A, with an edge directed from x to $f(x)$ for each element $x \in A$. This digraph will have the property that every vertex has out-degree equal to 1. Conversely, any such digraph is the graph of a unique function f, namely the function mapping each vertex x to the vertex at the other end of the unique edge directed out of x. Note that loops are allowed, since we may have elements with $f(x) = x$. A digraph in which every vertex has out-degree 1 is called a *functional digraph*.

In particular, if A has n elements, then there are n^n functions from A to itself, and hence n^n functional digraphs with vertex set A. If we regard these as structures on A, then they are enumerated by the exponential generating function

$$F(x) = \sum_n n^n \frac{x^n}{n!}.$$

This suggests a possible way of proving that the number of unordered rooted labelled trees on n vertices is given by $t(n) = n^{n-1}$, purely combinatorially, without using the Lagrange inversion formula. Namely, the identity

$$T(x) = \sum_{n=1}^{\infty} n^{n-1} \frac{x^n}{n!},$$

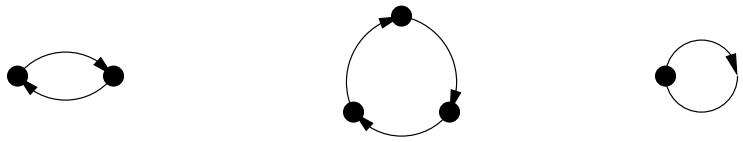
which we would like to prove, is equivalent to

$$xT'(x) = \sum_{n=1}^{\infty} n^n \frac{x^n}{n!} = F(x) - 1. \tag{1}$$

Here we subtracted 1 on the right hand side so as not to count the empty functional digraph. We must do this because the constant term on the left-hand side is zero.

We will use exponential generating function principles to find an identity satisfied by $F(x)$, and use this to prove that $F(x) = xT'(x) + 1$

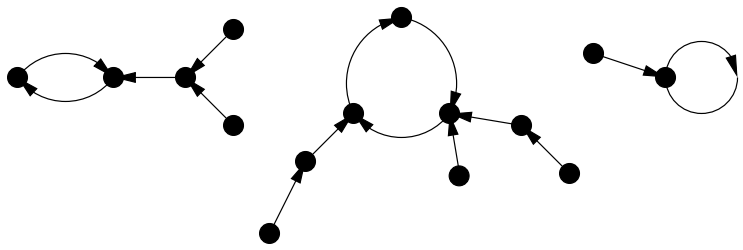
To get started, let us consider two special types of functional digraphs. The first type is the digraph of a permutation. Here the graph simply displays the cycle structure of the permutation, with the edges directed around each cycle, as indicated in the example shown here.



The second special type of functional digraph to consider is a rooted tree, with a loop on the root, and the rest of the edges directed toward the root. Then f is the function mapping each non-root vertex in the tree to its parent, and the root to itself.

Trees and permutations represent extreme cases of functional digraphs. At one extreme, in a permutation digraph, every vertex belongs to a cycle. At the other, in a tree, only one vertex, the root, belongs to a cycle. The general functional digraph can be described as a mix of permutations and trees. Suppose we start at a vertex x and follow directed edges from x to $f(x)$ to $f^2(x) = f(f(x))$ and so on. Eventually, since our vertex set is finite, we must return to a vertex already visited. At that point we fall into a cycle and we will continue around and around it forever. Consequently we can say this about the structure of a functional digraph: some vertices belong to cycles, and the rest of the vertices have paths leading to these vertices.

Now let x be a vertex in a cycle and consider the set X of all vertices y such that y does not belong to any cycle, and $f^k(y) = x$ for some k. Together with x, the vertices in X form a tree directed into x as the root, since in the graph on $X \cup \{x\}$, every path leads eventually to x. Now every vertex not belonging to a cycle belongs to one such tree for some vertex x that does belong to a cycle. In this way, given a functional digraph on a set A, we get a partition of A into rooted trees, and an arrangement of the roots of the trees into a bunch of cycles, that is, into a permutation. Here is a picture typical of the situation.



In this functional digraph, there are six trees, and the permutation on their roots is the one shown in the previous figure. Note that two of the trees in this example are one-element trees consisting of a root only.

What we have described is an equivalence of the structure "functional digraph" with the composite structure of (permutation)∘(rooted tree). Indeed, a directed rooted tree with all edges directed into the root is the same as an undirected rooted tree, since the tree itself determines the directions, and the structure of permutation that we have on the set of roots we can equally well think of as being on the set of trees themselves, with each tree represented by its root. The generating function for permutations is $\frac{1}{1-x} = 1/(1-x)$, and that for rooted trees is what we have denoted $T(x)$. Hence the generating function $F(x)$ for functional digraphs is related to $T(x)$ by

$$F(x) = 1/(1 - T(x)).$$

To prove (1) we have now only do to a little bit of calculus and algebra. Our starting point is the two identities

$$T(x) = xe^{T(x)}, \quad F(x) = 1/(1 - T(x)),$$

both of which we obtained directly from exponential generating function counting principles. Differentiating both sides of the first identity gives

$$T'(x) = e^{T(x)} + xe^{T(x)}T'(x),$$

and hence

$$xT'(x) = xe^{T(x)} + x(xe^{T(x)})T'(x).$$

Using $xe^{T(x)} = T(x)$, this simplifies to

$$xT'(X) = T(x) + xT(x)T'(x),$$

or

$$(1 - T(x))xT'(x) = T(x).$$

Therefore

$$xT'(x) = T(x)/(1 - T(x)) = 1/(1 - T(x)) - 1 = F(x) - 1,$$

which is what we wanted to show.

So also F is DA differentially algebraic.

Notes on cycle generating functions

1. The cycle generating function of a species

A *combinatorial species* is a rule attaching to each finite set X a set of structures on X, in such a way that the allowed structures do not depend on the particular names of the elements of X. This concept is best illustrated by giving examples of structures that do and do not form species.

In the notes on exponential generating functions, one kind of structure we enumerated was that of an *alternating permutation* on the numbers $1, 2, \ldots, n$. Recall that this meant a permutation $a_1, a_2 \ldots, a_n$ of these numbers such that $a_1 < a_2 > a_3 < a_4 \cdots$. These turned out to be interesting structures, because the exponential generating function for odd-length alternating permuations is $\tan x$, and the exponential generating function for even-length ones is $\sec x$. However, these structures do NOT belong to a species. The reason is that the condition $a_1 < a_2 > a_3 \cdots$ depends on the relation "<," which is specific to numbers. One way to express this difficulty is that there is no meaning to the concept of an alternating permutation of a set X whose elements have no built-in order. Another, somewhat more precise, way to formulate the trouble is that it is not possible to permute the numbers in an alternating permutation and still have an alternating permutation.

For constrast, consider the structure of *labelled tree* on a vertex set X. The concept of a labelled tree on any vertex set whatsoever is meaningful, because it does not depend on the particular names of the vertices. Again, we can formulate this more precisely by saying that you can permute the vertices of a labelled tree and get another (maybe the same) labelled tree. Labelled trees DO form a species.

Other examples of species are the species of all linear orderings of a set X, or the species of all permutations of a set X, or the trivial species which has just one structure on each set X. As we shall see, the linear orderings of X and the permuations of X are two *different* species, even though each has the same number of structures on an n element set, namely $n!$, and even though the two species therefore have the same exponential generating function. In fact, all of the examples of structures that we have enumerated using exponential generating functions are species, with the sole exception of alternating permutations.

If F is a species, we will denote the set of F-structures on a set X by $F(X)$. Because F is a species, the group of permutations of X acts on $F(X)$. In particular, the symmetric group S_n acts on $F([n])$. Here, as usual, $[n]$ is an abbreviation for $\{1, 2, \ldots, n\}$.

Definition. The *cycle generating function* of a species F is the formal power series in a variable x and infinitely many variables $p_1, p_2, \ldots$ defined by the formula

$$Z_F(p_1, p_2, \ldots; x) = \sum_{n=0}^{\infty} \frac{x^n}{n!} \sum_{g \in S_n} |F([n])^g| p_1^{j_1(g)} p_2^{j_2(g)} \cdots p_n^{j_n(g)}.$$

Here $F([n])^g$ denotes the set of F-structures on $[n]$ which are fixed by the permutation g, and $j_k(g)$ is the number of k-cycles of g (acting on $[n]$).

To make things clear, let's work out the first few terms of Z_F for the species F of labelled, unrooted trees. For each n and each permutation $g \in S_n$, we need to figure out how many trees are fixed by g. It will be enough to do this for one permutation of each cycle type, since they all fix the same number of trees.

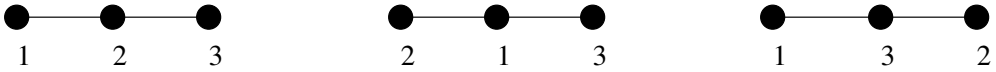
For $n = 0$, there is just one tree, and just one g, namely the identity permutation of the empty set, which has no cycles. The corresponding term in Z_F is 1.

For $n = 1$, there is again just one tree and one permutation, the identity, but now it has a single cycle of length 1, so its monomial is p_1. The corresponding term in Z_F is xp_1.

For $n = 2$, there is still just one tree. Now there are two permutations, the identity and (1 2), and of course they both fix the one tree. The corresponding term in Z_F is

$$\frac{x^2}{2}(p_1^2 + p_2).$$

For $n = 3$, there are three trees, shown here.



The identity element in S_3 obviously fixes all of them, contributing $3p_1^3$. The element $g = (1\ 2)$ fixes one tree, the last one shown above, contributing a monomial p_1p_2. There are two more transpositions, (1 3) and (2 3), contributing a total of $3p_1p_2$. The 3-cycles (1 2 3) and (1 3 2) do not fix any trees, so they contribute nothing. The resulting third term in Z_F is

$$\frac{x^3}{6}(3p_1^3 + 3p_1p_2) = \frac{x^3}{2}(p_1^3 + p_1p_2).$$

Before we compute the fourth term, let us notice something interesting about the third term. The *automorphism group* $\mathrm{Aut}(T)$ of a tree T on labelled vertices $\{1, \ldots, n\}$ is defined to be the set of permutations $g \in S_n$ that fix T. Informally speaking, these are the symmetries of T. The automorphism group of any F-structure on a set X is defined similarly. So, for instance, the automorphism group of the last tree shown above is the two-element subgroup $G = \{(1)(2)(3), (1\ 2)(3)\}$ of S_3.

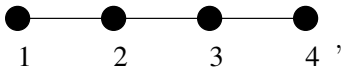
Given any group G of permutations, we may assign each element $g \in G$ weights $j_1(g), j_2(g), \ldots$, where $j_k(g)$ is the the number of k-cycles in g, as before, and form the ordinary generating function $\sum_{g\in G} p_1^{j_1(g)} p_2^{j_2(g)} \cdots p_n^{j_n(g)}$. The *cycle index* of G is defined to be $1/|G|$ times this generating function. In the example above, with $G = \{(1)(2)(3), (1\ 2)(3)\}$, the cycle index is

$$C_G(p_1, p_2, \ldots) = \frac{1}{2}(p_1^3 + p_1p_2).$$

Notice that the third term in Z_F, which we just computed, is equal to x^3 times this cycle index. In this case we got the group G from the last tree, but since the others are isomorphic to it, their automorphism groups have the same cycle index, although they are different groups.

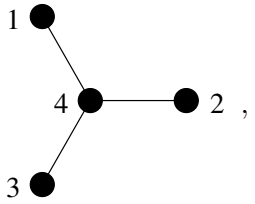
All this suggests that it might be more efficient to evaluate the terms of Z_F, not by considering each permutation g and looking at the trees that it fixes, but by considering each isomorphism type of tree, and looking at the permutations that fix it, that is, at its automorphism group. Let's take this approach to computing the fourth term of the cycle generating function Z_F for trees.

There are two isomorphism types of trees on four vertices. A typical tree of the first type is a "path"



,

and there are 12 trees isomorphic to and including this one. A typical tree of the second type is a "star"



,

and there are 4 trees of this type. A "path" tree has two automorphisms: the automorphism group of the one shown above consists of the identity and the permutation $g = (1\ 4)(2\ 3)$. Each of the

twelve "path" trees therefore contributes $p_1^4 + p_2^2$ to the total $\sum_{g \in S_4} |(\text{trees})^g| p_1^{j_1}(g) p_2^{j_2(g)} \cdots$, for a net contribution of

$$12(p_1^4 + p_2^2).$$

The automorphism group of a "star" tree is the symmetric group on the labels of the three outer vertices, the center vertex being always fixed. For the one shown above it is the subgroup of S_4 which is a copy of S_3, with all its elements fixing 4. Each "star" tree therefore contributes $p_1^4 + 3p_1^2 p_2 + 2p_1 p_3$, for a net contribution of

$$4(p_1^4 + 3p_1^2 p_2 + 2p_1 p_3).$$

Hence the fourth term of Z_F is

$$x^4 \left(\frac{1}{2}(p_1^4 + p_2^2) + \frac{1}{6}(p_1^4 + 3p_1^2 p_2 + 2p_1 p_3) \right).$$

Note that this is x^4 times the sum of the cycle indices for the automorphism groups of the two types of trees. This is not an accident; we will prove the general result below.

Returning to our example and writing out all terms computed so far, we have for the species F of labelled trees,

$$Z_F = 1 + x p_1 + \frac{x^2}{2}(p_1^2 + p_2) + \frac{x^3}{6}(3p_1^3 + 3p_1 p_2) + \frac{x^4}{24}(16p_1^4 + 12p_2^2 + 12p_1^2 p_2 + 8p_1 p_3) + \cdots .$$

Let's make a couple of additional observations about this result. First of all, every tree is obviously fixed by the identity element, so the term involving p_1^n has coefficient equal to the number of labelled trees, which we know should be n^{n-2}. This agrees with the coefficients you see above: 1, 1, 1, 3, 16. We can extract just these terms by setting $p_1 = 1$ and every other $p_k = 0$. Then we get

$$Z_F(1, 0, 0, \ldots; x) = 1 + x + \frac{x^2}{2} + 3\frac{x^3}{6} + 16\frac{x^4}{24} + \cdots ,$$

which is the usual exponential generating function for labelled trees.

Second, suppose we set every $p_k = 1$. Then the cycle index of each automorphism group becomes 1, and we get

$$Z_F(1, 1, 1, \ldots; x) = 1 + x + x^2 + x^3 + 2x^4 + \cdots ,$$

which is the *ordinary* generating function for *unlabelled* trees. Here you begin to see why exponential generating functions count labelled structures and ordinary generating functions count unlabelled ones: each kind of generating function is a special value of the cycle generating function, which incorporates them both.

2. Properties of the cycle generating function

We now turn to some general properties of the cycle generating function $Z_F(p_1, p_2, \ldots; x)$. We have already seen some of these properties in the example in Section 1. To establish the main properties, we will look at the formula for Z_F from two different vantage points.

First of all, we can view Z_F as the exponential generating function for the sums

$$\sum_{g \in S_n} |F([n])^g| p_1^{j_1(g)} p_2^{j_2(g)} \cdots p_n^{j_n(g)}.$$

This sum in turn is an ordinary generating function: a weighted enumerator for pairs (f, g) consisting of an F-structure f and a permutation g that fixes f, with the weight of the pair keeping track of the cycle structure of g, and given by $p_1^{j_1(g)} p_2^{j_2(g)} \cdots p_n^{j_n(g)}$. Hence Z_F is the mixed exponential/ordinary generating function for structures consisting of a pair (f, g), weighted by the cycle monomial of g. Now if we set $p_1 = 1$ and $p_k = 0$ for all other k, we are setting the weight to

zero for all pairs in which g is not the identity element, and to 1 for pairs $(f, 1)$. This gives the exponential generating function for the structures f, since to choose a pair $(f, 1)$ is just to choose f. We summarize these observations as a theorem.

Theorem 1. *The special value*

$$Z_F(1, 0, 0, \ldots; x)$$

of the cycle generating function for a species F is equal to the usual exponential generating function for F-structures.

Next, let us rewrite the formula for Z_F by summing first over structures f, and then over permutations g that fix f, that is, over $g \in \mathrm{Aut}(f)$. We have

$$Z_F = \sum_n \frac{x^n}{n!} \sum_{f \in F([n])} \sum_{g \in \mathrm{Aut}(f)} p_1^{j_1(g)} p_2^{j_2(g)} \cdots p_n^{j_n(g)}.$$

Now for different isomorphic structures f, that is, for different structures f in the same orbit of the action of S_n on $F([n])$, the sum $\sum_{g \in \mathrm{Aut}(f)} p_1^{j_1(g)} p_2^{j_2(g)} \cdots p_n^{j_n(g)}$ is the same. For each orbit of F-structures, this sum appears as many times as the size of the orbit. By the orbit-stabilizer theorem, the size of the orbit is given by $|\,\mathrm{orb(f)}\,| = |S_n|/|\,\mathrm{Aut}(f)|$, since $\mathrm{Aut}(f)$ is, by definition, the stabilizer of f in S_n. Hence we have

$$Z_F = \sum_n \frac{x^n}{n!} \sum_{\mathrm{Orb}(S_n, F([n]))} \frac{n!}{|\,\mathrm{Aut}(f)|} \sum_{g \in \mathrm{Aut}(f)} p_1^{j_1(g)} p_2^{j_2(g)} \cdots p_n^{j_n(g)},$$

which is equal to

$$Z_F = \sum_n x^n \sum_{\mathrm{Orb}(S_n, F([n]))} C_{\mathrm{Aut}(f)}(p_1, p_2, \ldots).$$

Here $\mathrm{Orb}(S_n, F([n]))$ denotes the set of orbits of S_n acting on the F-structures on $[n]$, and $C_{\mathrm{Aut}(f)}(p_1, p_2, \ldots)$ is the cycle index of the automorphism group of any structure f belonging to the chosen orbit.

Now an orbit of S_n on $F([n])$ is the same thing as an *unlabelled* F-structure, so we have arrived at a new interpretation of Z_F: it is the *ordinary* generating function for the cycle indices of the automorphism groups of *unlabelled* F-structures. From this we deduce a second theorem giving a special value of Z_F.

Theorem 2. *The special value*

$$Z_F(1, 1, 1, \ldots; x)$$

of the cycle generating function for a species F is equal to the ordinary generating function for unlabelled *F-structures.*

Proof. This follows from the preceding observations and the fact that for any group of permutations G, the cycle index satisfies $C_G(1, 1, 1, \ldots) = 1$. The latter fact holds because $C_G(p_1, p_2, \ldots)$ is by definition $1/|G|$ times a sum of $|G|$ monomials in the variables p_k. □

3. More examples

In this section we will compute the cycle generating functions for some of the most important species.

Example. Let F be the *trivial* species, with only one F-structure on $[n]$ for every n. Then of course there is only one unlabelled F-structure, and its automorphism group is all of S_n. It follows

that the cycle generating function for the trivial species is

$$Z_{\text{triv}} = \sum_{n=0}^{\infty} x^n C_{S_n}(p_1, p_2, \ldots), \tag{1}$$

the ordinary generating function for the cycle indices of the symmetric groups S_n. Of course this is only useful if we can calculate $C_{S_n}(p_1, p_2, \ldots)$. However, Z_{triv} can also be viewed as a mixed ordinary/exponential generating function for permutations, in which each permutation counts with a weight monomial $p_1^{j_1} p_2^{j_2} \cdots$ for a permutation with j_1 1-cycles, j_2 2-cycles, and so on. Without this extra detail, we would count permutations as composite structures (trivial) $\circ$ (cycle), giving the exponential generating function

$$e^{-\log(1-x)} = e^{x+x^2/2+x^3/3+\cdots}.$$

But it is easy to modify this to keep track of the cycle lengths: in the k-th term of the exponent, we just introduce a factor p_k. Hence we have the formula

$$Z_{\text{triv}} = e^{p_1 x + p_2 x^2/2 + p_3 x^3/3 + \cdots}. \tag{2}$$

By expanding and comparing term by term with equation (1) above, we can use this formula to compute the cycle indices $C_{S_n}(p_1, p_2, \ldots)$ for arbitrary n.

Let's look briefly at the special values given by the theorems in Section 2. First, if we set $p_1 = 1$ and all other $p_k = 0$, we get

$$Z_{\text{triv}}(1, 0, 0, \ldots; x) = e^x.$$

This agrees with the familiar exponential generating function for the trivial species. Next, if we set all $p_k = 1$, we get

$$e^{-\log(1-x)} = \frac{1}{1-x}.$$

This is the *ordinary* generating function for the *unlabelled* trivial species. There is one trivial unlabelled structure for each n, and we have obtained, as we expect, the geometric series with all coefficients equal to 1.

Example: The species of *linear orderings*. The automorphism group of a linear ordering is trivial, since any permutation of the elements changes the order. All the linear orderings of $[n]$ form a single S_n orbit, which is to say, there is just one unlabelled linear ordering for each n. Using the interpretation of Z_F as an ordinary generating function for cycle indices of automorphism groups, we immediately obtain the formula

$$Z_{\text{lin}} = \sum_{n=0}^{\infty} x^n p_1^n = \frac{1}{1 - p_1 x}.$$

The first special value is

$$Z_{\text{lin}}(1, 0, 0, \ldots; x) = \frac{1}{1-x},$$

which is the exponential generating function for linear orderings. The second special value is

$$Z_{\text{lin}}(1, 1, 1, \ldots; x) = \frac{1}{1-x}.$$

Of course this comes out to the same thing because Z_{lin} only depends on p_1 and x. However, the result is now to be interpreted as the ordinary generating function for unlabelled linear orderings. Since there is just one of these for each n, we get the geometric series.

Example: The species of *permutations*. Here we must be clear about how S_n acts on the set of permutations of $[n]$, which is to say, on itself. It acts by permuting the numbers in the cycle

notation for a permutation. In group-theoretic terms, this means that g sends σ to $g\sigma g^{-1}$. In other words, S_n acts on itself by conjugation. Now let's see how many permutations σ are fixed by g. We have

$$g\sigma g^{-1} = \sigma \quad \text{if and only if} \quad g\sigma = \sigma g,$$

that is, if and only if σ commutes with g. Note that in particular, g fixes σ if and only if σ fixes g. This is an extremely useful fact. For the number of elements σ fixed by g is equal to the number that fix g, that is, to the stabilizer of g in the action of S_n on itself by conjugation. By the orbit-stablizer theorem, this number is equal to $n!/|C(g)|$, where $C(g)$ is the conjugacy class of g, consisting of all permutations with the same cycle structure as g. Using this, we find

$$Z_{\text{perm}} = \sum_{n=0}^{\infty} \frac{x^n}{n!} \sum_{g \in S_n} \frac{n!}{|C(g)|} p_1^{j_1(g)} p_2^{j_2(g)} \dots p_n^{j_n(g)}.$$

In this formula, the $n!$ in numerator and denominator cancel, and $|C(g)|$ is exactly the number of terms in the sum with a given monomial $p_1^{j_1(g)} p_2^{j_2(g)} \dots p_n^{j_n(g)}$. Hence each such monomial occurs with a net coefficient exactly equal to x^n. In short, we have

$$Z_{\text{perm}} = \sum_{n=0}^{\infty} x^n \sum_{|\lambda|=n} p_{\lambda_1} p_{\lambda_2} \cdots p_{\lambda_l}.$$

This is the ordinary generating function for all partitions λ, counted with weight $x^n p_1^{j_1} p_2^{j_2} \cdots$, where n is the size of λ and j_k is the number of parts equal to k in λ. Using our familiar techniques for partition enumeration, we can rewrite this as

$$Z_{\text{perm}} = \prod_{k=1}^{\infty} \frac{1}{1 - p_k x^k}.$$

The first special value is

$$Z_{\text{perm}}(1, 0, 0, \dots; x) = \frac{1}{1-x}.$$

This is the exponential generating function for permutations. As expected it is the same as the exponential generating function for linear orderings. Note, however, that the full cycle generating functions Z_{lin} and Z_{perm} are not the same. This shows very clearly the important point that linear orderings and permutattions are *two different species*.

The second special value is

$$Z_{\text{perm}}(1, 1, 1, \dots; x) = \prod_{k} \frac{1}{1 - x^k}.$$

This is the familiar ordinary generating function for partitions. It should also be the ordinary generating function for unlabelled permutations. Is this correct? Well, an "unlabelled" permutation is entirely determined by its cycle structure, that is, two permutations are in the same orbit if they have the same cycle lengths. The cycle lengths form a partition of n, so unlabelled permutations are nothing but partitions, and the formula checks.

4. Decorated F-structures

We have seen that the two special values $Z_F(1, 0, 0, \dots; x)$ and $Z_F(1, 1, 1, \dots; x)$ have enumerative meaning, but what about the full cycle generating function itself? The answer to this comes from the Polya-Redfield theorem. Remember that Z_F is the ordinary generating function for the cycle indices of automorphism groups of unlabelled structures. Now suppose we have a set of "colors"

A, with some symbolic weights assigned, and enumerated by an ordinary generating function $G(y)$. We use the notation $p_k[G(y)] = G(y^k)$. Then the Polya-Redfield theorem tells us that

$$C_{\mathrm{Aut}(f)}[G(y)] \underset{\text{def}}{=} C_{\mathrm{Aut}(f)}(G(y), G(y^2), \ldots)$$

is the ordinary generating function for C-colorings of the set $[n]$, up to symmetries given by the automorphism group of f. In other words, it is the ordinary generating function for unlabelled structures isomorphic to f, with their vertices "decorated" with colors from the set A. Summing this over all F-structures, with a factor x^n to keep track of the size of the vertex set, gives

$$Z_F[G(y)] = Z_F(G(y), G(y^2), \ldots; x),$$

and this is the ordinary generating function for all unlabelled vertex-decorated F-structures, with a weight $x^n y^k$ if it has n vertices and the coloring contributes weight y^k. More generally, the generating function for the colors can be a weight enumerator $G(y_1, y_2, \ldots, y_r)$ in any number of variables, and then for p_k we substitute $p_k[G(\mathbf{y})] = G(y_1^k, y_2^k, \ldots, y_r^k)$.

The simplest possibility, but also in some ways the most general, is that the set A consists of r colors each with its own symbolic variable y_i. Then the enumerator of the colors is $y_1 + y_2 + \cdots + y_r$, and

$$p_k[y_1 + y_2 + \cdots + y_r] = y_1^k + y_2^k + \cdots + y_r^k.$$

This quantity is called the k-th *power-sum* of the variables y_i. This is the reason I have used the letter p for the variables p_k, to stand for "power-sum." We can summarize the above considerations as a theorem.

Theorem 3. *The special value*

$$Z_F[y_1 + y_2 + \cdots] = Z_F(p_1(\mathbf{y}), p_2(\mathbf{y}), \ldots; x),$$

where $p_k(\mathbf{y}) = p_k[y_1 + y_2 + \cdots]$ is the k-th power-sum, is the ordinary generating function for unlabelled F-structures decorated with colors on the vertices, and enumerated with weight $x^n y_1^{c_1} y_2^{c_2} \cdots$, where n is the number of vertices and c_i is the number that receive color i.

An important point about this theorem is that the "special" value involved is actually a fully general value. In the theory of symmetric functions one shows that when there are infinitely many variables y_i, the power-sums $p_k(\mathbf{y})$ are in effect independent variables. The interpretation of Z_F as an ordinary generating function for decorated unlabelled F-structures therefore involves no loss of information.

Example: If we take the trivial structure on a set of n elements, decorate the elements with colors $1, \ldots, r$, and consider this as an unlabelled structure, we have a distribution of n identical items to r distinct recipients. This is just a multiset of n elements from the r colors. We assign color i a variable y_i. Then the ordinary generating function for multisets of the colors, counted with a monomial $x^n y_1^{k_1} \cdots y_r^{k_r}$, where n is the size of the multiset and k_i is the number of copies of color i in it, should be given by

$$Z_{\mathrm{triv}}(p_1(\mathbf{y}), p_2(\mathbf{y}), \ldots; x).$$

Using formula (2) for Z_{triv}, we see that this is equal to

$$\Omega[x(y_1 + \cdots + y_r)],$$

where we define

$$\Omega = e^{p_1 + p_2/2 + p_3/3 + \cdots},$$

and the square bracket notation means as usual that p_k goes to $p_k[x(y_1+\cdots+y_r)] = x^k(y_1^k+\cdots+y_r^k) = x^k p_k(\mathbf{y})$. Now notice that in general we have $p_k[X+Y] = p_k[X]+p_k[Y]$ for any quantities X, Y. Hence the exponent in the definition of Ω is additive, and Ω itself is multiplicative:

$$\Omega[X+Y] = \Omega[X]\Omega[Y].$$

We can use this compute $\Omega[xy_1+\cdots+xy_r]$ by computing $\Omega[xy_i]$ and multiplying. However

$$\Omega[xy_i] = e^{-\log(1-xy_i)} = \frac{1}{1-xy_i},$$

so

$$\Omega[x(y_1+\cdots+y_r)] = \prod_{i=1}^{r} \frac{1}{1-xy_i}.$$

This agrees with the generating function for multisets that we would get by familiar ordinary generating function methods.

5. Plethysm

Our final topic will be the analog for cycle generating functions of the product and composition principles for exponential generating functions. It turns out that the stray variable x in the cycle generating function is an unnecessary nuisance and it is best to get rid of it before proceeding further. We define the *reduced* cycle generating function for a species F to be

$$\widehat{Z}_F(p_1,p_2,\ldots) = Z_F(p_1,p_2,\ldots;1).$$

There is no real loss of information in working with $\widehat{Z}_F$ instead of Z_F. To see this, note that in the x^n term of Z_F, every monomial $p_1^{j_1(g)}p_2^{j_2(g)}\ldots p_n^{j_n(g)}$ has degree n, if we agree to consider p_k as having degree k. If we substitute $p_k \mapsto x^k p^k$ in such a monomial, the effect is to multiply it by x^n. This shows that

$$Z_F(p_1,p_2,\ldots;x) = \widehat{Z}_F(xp_1,x^2p_2,\ldots),$$

so we can always recover Z_F if we know $\widehat{Z}_F$.

The product principle for cycle generating functions is the same as the familiar one for exponential generating functions.

Theorem 4. *If $F = GH$ is a product structure, then*

$$\widehat{Z}_F = \widehat{Z}_G\widehat{Z}_H$$

(and hence also $Z_F = Z_G Z_H$).

Proof. Recall that a GH structure on $[n]$ consists of an ordered partition of $[n]$ into subsets A_1 and A_2, with a G structure on A_1 and an H structure on A_2. To decorate the GH structure is the same thing as to decorate the G structure on A_1 and the H structure on A_2 independently. When we drop the labels, that is, look at S_n orbits, it no longer matters which elements belong to A_1 and which to A_2: an unlabelled decorated GH structure just consists of an unlabelled decorated G structure and an unlabelled decorated H structure. These can be chosen independently, and the product principle is now reduced to the usual one for ordinary generating functions.

What this actually shows is that

$$\widehat{Z}_F[Y] = \widehat{Z}_G[Y]\widehat{Z}_H[Y]$$

for any Y. But if we take $Y = y_1+y_2+\cdots$, then $p_k[Y] = p_k(\mathbf{y})$, and these power-sums are algebraically independent for all k. Hence the identity evaluated at Y implies the full identity that we wanted to prove. □

An interesting feature of this proof is that by using the interpretation of Z_F as an ordinary generating function for decorated unlabelled structures, we have reduced the product principle to the one for ordinary generating functions. At the same time, since the exponential generating function for F structures is a special value of Z_F, this result implies the product principle for exponential generating functions.

To give the cycle generating function version of the composition principle, we define a new operation called plethysm.

Definition Let G and H be formal power series in the variables p_1, p_2,..., and assume that the constant term of H is zero. The *plethysm* $G * H$ of H into G is defined to by

$$G * H = G(p_1 * H, p_2 * H, \ldots),$$

where

$$p_k * H = H(p_k, p_{2k}, p_{3k}, \ldots)$$

In other words, to plethystically substitute H into G, we substitute $p_k * H$ for each variable p_k in G, and to compute $p_k * H$, we substitute p_{jk} for each variable p_j in H.

Lemma 1. *The plethysm is related to the bracket operation by the identity*

$$G[H[Y]] = (G * H)[Y].$$

Proof. To compute $H[Y]$ we replace each variable p_j in H by $p_j[Y]$, which is defined as the result of replacing every symbol in Y by its j-th power. To compute $G[H[Y]]$ we replace each variable p_k in G by $p_k[H[Y]]$. This is obtained by replacing every symbol in $H[Y]$ by its k-th power. However, we could have gotten the same result by replacing each variable p_j in H by $p_{jk}[Y]$ in the first place. In other words, $p_k[H[Y]] = (p_k * H)[Y]$ according to our definition of $p_k * H$. Now we substitute this for each p_k in G to obtain $(G * H)[Y]$. □

Of course, the reason we defined plethysm as we did was to make this lemma work. Thus the lemma is largely just a matter of notation. Now we are ready for a real theorem.

Theorem 5. *If $F = G \circ H$ is a composite structure, then*

$$\widehat{Z}_F = \widehat{Z}_G * \widehat{Z}_H.$$

Proof. We will prove that

$$\widehat{Z}_F[Y] = (\widehat{Z}_G * \widehat{Z}_H)[Y]$$

for any Y. This is sufficient to prove the general identity, for the same reason as in the proof of the product principle. By the Lemma, the right-hand side above is equal to

$$\widehat{Z}_G[\widehat{Z}_H[Y]].$$

We must verify that this is a generating function for the same kind of decorated structures as $\widehat{Z}_F[Y]$.

We can interpret $\widehat{Z}_G[\widehat{Z}_H[Y]]$ as the generating function for unlabelled G structures with their vertices decorated by "colors" whose ordinary generating function is $\widehat{Z}_H[Y]$. However, the latter is the generating function for decorated unlabelled H-structures. Therefore $\widehat{Z}_G[\widehat{Z}_H[Y]]$ enumerates structures consisting of an unlabelled G-structure on a set whose elements are in turn decorated with unlabelled decorated H-structures.

Now let's see what an unlabelled decorated $G \circ H$ structure on a set X looks like. We are to choose a partition Π of X, a an H-structure on each block of Π, and a G-structure on the set of blocks of Π. Moreover, we are to decorate the vertices of X, which is the same thing as decorating the vertices of the various H-structures in the composite structure independently. Removing labels,

we are left with a multiset of unlabelled decorated H-structures, with a G-structure on the elements of the multiset. But this is the same thing as an unlabelled G-structure on elements decorated by unlabelled decorated H-structures, which is what $\widehat{Z}_G[\widehat{Z}_H[Y]]$ enumerates. □

Example: The cycle generating function for the species of set partitions can be computed using plethysm. Our species is a composite (trivial) ∘ (non-empty set). Previously, we used this to get the exponential generating function e^{e^x-1} for set partitions. Here, in place of e^x we need to use the cycle generating function for a trivial species, and in place of functional composition, we need to use plethysm.

In the notation introduced above, we have

$$\widehat{Z}_{\text{triv}} = \Omega = e^{p_1+p_2/2+\cdots},$$

and subtracting the constant term, which counts the trivial structure on the empty set, we have

$$\widehat{Z}_{\text{non-empty}} = \Omega - 1.$$

Hence

$$\widehat{Z}_{\text{partition}} = \Omega * (\Omega - 1).$$

Let's compute some terms of this, keeping track of everything up to degree 4. We have

$$\begin{aligned}
p_1 * (\Omega - 1) &= \Omega - 1 = e^{p_1+p_2/2+p_3/3+p_4/4+\cdots} - 1 \\
p_2 * (\Omega - 1) &= e^{p_2+p_4/2+\cdots} - 1 \\
p_3 * (\Omega - 1) &= e^{p_3+\cdots} - 1 \\
p_4 * (\Omega - 1) &= e^{p_4+\cdots} - 1.
\end{aligned}$$

Then

$$\begin{aligned}
\Omega * (\Omega - 1) &= e^{p_1*(\Omega-1)+p_2*(\Omega-1)/2+p_3*(\Omega-1)/3+p_4*(\Omega-1)/4+\cdots} \\
&= e^{p_1*(\Omega-1)} e^{p_2*(\Omega-1)/2} e^{p_3*(\Omega-1)/3} e^{p_4*(\Omega-1)/4} \cdots .
\end{aligned}$$

Expanding the various exponentials, substituting for $p_k * (\Omega - 1)$ from the chart above, collecting terms, and putting back the x^n factors for clarity, we find after some work:

$$\begin{aligned}
Z_{\text{partition}}(p_1, p_2, \ldots; x) = 1 &+ x\, p_1 + \frac{x^2}{2}\left(2\, {p_1}^2 + 2\, p_2\right) + \frac{x^3}{6}\left(5\, {p_1}^3 + 9\, p_1\, p_2 + 4\, p_3\right) \\
&+ \frac{x^4}{24}\left(15\, {p_1}^4 + 42\, {p_1}^2\, p_2 + 21\, {p_2}^2 + 24\, p_1\, p_3 + 18\, p_4\right) + \cdots .
\end{aligned}$$

Looking at the x^3 term, this tells us for instance that there are 5 labelled partitions of a 3 element set (the coefficient of the term involving p_1^3), and 3 unlabelled ones (setting all $p_k = 1$ in the x^3 term). You can easily check by hand that this is correct.

Now let's apply our two standard specializations to this. Since we are using reduced cycle generating functions, in order to recover the exponential generating function for set partitions we should set $p_1 = x$ and $p_k = 0$ for all $k > 1$. You can check from the definition of plethysm that making this substitution in $F * G$ is equivalent to first applying it to each of F and G, then composing the resulting functions $F(x)$ and $G(x)$. In other words, the plethysm principle for cycle generating functions reduces under this substittution to the composition principle for exponential generating functions. In particular, our cycle generating function $\Omega * (\Omega - 1)$ for set partitions reduces to the exponential generating function e^{e^x-1} that we knew before.

On the other hand, to get the ordinary generating function for unlabelled structures we want to set $p_k = x^k$. From the definitions we see that this substitution sends $F * G$ to $F(G(x), G(x^2), \ldots)$, where $G(x)$ is shorthand for $G(x, x^2, \ldots)$, that is, for the ordinary generating function for the

unlabelled structures counted by G. In the present example, $\Omega - 1$ is the cycle generating function for the trivial non-empty species, so setting $p_k = x^k$ in $\Omega - 1$ gives the ordinary generating function $G(x) = x/(1-x)$ for the unlabelled trivial non-empty species.

Now let's calculate

$$\Omega(G(x), G(x^2), \ldots) = e^{x/(1-x)+(1/2)x^2/(1-x^2)+\cdots}.$$

The expression in the exponent is

$$\begin{aligned}\sum_{n=1}^{\infty} \frac{1}{n} \sum_{k=1}^{\infty} x^{kn} &= \sum_{k=1}^{\infty w} \sum_{n=1}^{\infty} \frac{x^{kn}}{n} \\ &= \sum_{k=1}^{\infty} \ln\left(\frac{1}{1-x^k}\right).\end{aligned}$$

Exponentiating this gives

$$\Omega(G(x), G(x^2), \ldots) = \prod_{k=1}^{\infty} \frac{1}{1-x^k}$$

as the ordinary generating function for unlabelled set partitions. Since an unlabelled partition of an n element set is merely an integer partition of n, this agrees with the ordinary generating function for integer partitions, as expected.