GENERATING FUNCTIONS

1. Combinatorial Families

A *combinatorial family* (or *family of combinatorial objects*) is a collection $\mathcal{A}$ of objects together with a function size : $\mathcal{A} \to \mathbb{N}_0$ (where $\mathbb{N}_0$ denotes the set of non-negative integers). In other words, a combinatorial family is just a collection of objects in which each object has some "size" associated with it, this being zero or a positive integer.

Example 1.1

Let $\mathcal{B}$ be the family of all (finite) binary strings, and for any string β in this family, let size(β) be the length of the string β.

$\mathcal{B} = \{\epsilon, 0, 1, 00, 01, 10, 11, 000, 001, 010, 011, 100, 101, 110, 111, \ldots\}$

(where ϵ denotes the empty string).

Example 1.2

Let S be a finite set, and let $\mathcal{P}$ be the collection of all subsets of S. Then $\mathcal{P}$ forms a combinatorial family with the size of each object X (a subset of S) being its cardinality $|X|$ (the number of elements in the subset X).

Given a combinatorial family $\mathcal{A}$, we denote by $\mathcal{A}_n$ the **collection of all objects of $\mathcal{A}$ with size** n (for any non-negative integer n). The *number of objects* in $\mathcal{A}_n$ is denoted by a_n.

In Example 1.1, $\mathcal{B}_n$ is the set of all binary strings of length n, and b_n (the number of binary strings of length n) is 2^n. For instance, $\mathcal{B}_0 = \{\epsilon\}$, $\mathcal{B}_1 = \{0, 1\}$, $\mathcal{B}_2 = \{00, 01, 10, 11\}$.

In Example 1.2, let m be the cardinality of the finite set S. Since $\mathcal{P}$ is the family formed by the *set of all subsets of S with the cardinality of a subset as its size*, it is obvious that $\mathcal{P}_n$ is the collection of *all subsets of S that have cardinality n*. As there are $\binom{m}{n}$

ways of selecting n elements out of m (i.e., of forming a subset of S with exactly n elements), we see that $p_n = \binom{m}{n}$.

1.1. **Basic Operations on Combinatorial Families.** Given two combinatorial families $\mathcal{A}$ and $\mathcal{B}$, there are two simple ways to construct a new family using these. A combinatorial family being nothing more than a collection of objects with well-defined sizes, constructing a new family boils down to constructing new objects from given objects. An obvious way to do this is to take two objects and simply put them together (or juxtapose them). Another (perhaps less obvious) way to do this is two take two objects and keep only one of them, discarding the other.

Definition 1.1

The *product* of two families $\mathcal{A}$ and $\mathcal{B}$ is the family $\mathcal{C} = \mathcal{A}\mathcal{B}$ consisting of all possible pairs (α, β), where α is an object of $\mathcal{A}$ and β an object of $\mathcal{B}$. The size function of $\mathcal{C}$ is defined as $\text{size}(\alpha, \beta) = \text{size}(\alpha) + \text{size}(\beta)$ (where $\text{size}(\alpha)$ and $\text{size}(\beta)$ are the sizes of the objects α of $\mathcal{A}$ and β of $\mathcal{B}$ respectively).

It should be clear that this definition is analogous to the notion of Cartesian product of two sets. Also note that we can define the product of more than two families by repeated application of this product.

Definition 1.2

The *disjoint union* of two families $\mathcal{A}$ and $\mathcal{B}$ is the family $\mathcal{C} = \mathcal{A} + \mathcal{B}$ in which each object γ is either an object α of $\mathcal{A}$ or an object β of $\mathcal{B}$. The size function of $\mathcal{C}$ is defined as

$$\text{size}(\gamma) = \begin{cases} \text{size}(\alpha), & \gamma = \alpha \\ \text{size}(\beta), & \gamma = \beta. \end{cases}$$

In this definition, even when two objects of $\mathcal{A}$ and $\mathcal{B}$ are the same, they are distinct objects of $\mathcal{A} + \mathcal{B}$. So this operation is analogous not to the union of sets, but rather the *disjoint union* of sets.

Example 1.3

If S and T are two finite sets with no common elements, we can define the combinatorial families $\mathcal{P}$ and $\mathcal{Q}$ formed by the respective collections of all subsets of each set, as given in Example 1.2.

(1) What will the product $\mathcal{PQ}$ of these two families represent? Each object of $\mathcal{PQ}$ is a pair of the form (X, Y), where X is an object of $\mathcal{P}$, i.e., $X \subseteq S$, and similarly, $Y \subseteq T$. We may think of this pair of objects as just being the subset $X \cup Y$ of $S \cup T$. This is valid because $\text{size}(X, Y) = \text{size}(X) + \text{size}(Y) = |X| + |Y|$.

(2) What about the disjoint union $\mathcal{P} + \mathcal{Q}$? This is simply the collection of all subsets of S and T (note: **not** subsets of $S \cup T$).

Exercises.

1.1 In Example 1.2, if S has m elements, how many objects does the family $\mathcal{P}$ have? Do you get the same answer if you compute it as $\sum p_n$?

1.2 In Example 1.3, if S has s elements and T has t elements, how many objects do the following families contain?

(i) $\mathcal{P}$ and $\mathcal{Q}$

(ii) $\mathcal{PQ}$

(iii) $\mathcal{P} + \mathcal{Q}$

1.3 Let $\mathcal{E}$ be the family consisting of all binary strings containing an even number of 1s, and $\mathcal{O}$ the family of binary strings containing an odd number of 1s. In each case, define the size of a string to be its length. Determine e_n and o_n for all n.

1.4 For the families $\mathcal{E}$ and $\mathcal{O}$ defined in Exercise 1.3, let $\mathcal{B} = \mathcal{E} + \mathcal{O}$. Compute b_n for all n. What is a simple description of the family $\mathcal{B}$?

1.5 Let $\mathcal{T}$ be the family containing only one object – the binary string 1 with $\text{size}(1) = 1$. Similarly, let $\mathcal{F}$ be the family containing only the binary string 0 of length 1. Show that the family $\mathcal{ET} + \mathcal{OF}$ is equivalent to the family $\mathcal{O}$ (where $\mathcal{E}$ and $\mathcal{O}$ are as defined in Exercise 1.3).

1.6 Given two families $\mathcal{A}$ and $\mathcal{B}$, let $C = \mathcal{A} + \mathcal{B}$ and $D = \mathcal{AB}$. Find the values of c_n and d_n (in terms of a_n and b_n).

2. The Generating Function of a Combinatorial Family

In the previous section, we saw two different ways to build new combinatorial families in terms of known ones using the product and disjoint union operations. We also saw how it is possible for such a new family constructed using these operations to be equivalent to one of the operands itself (Exercise 1.5). Now we shall define a powerful tool that helps us count various collections of objects in combinatorial families.

Definition 2.1

The *generating function* $g(x)$ of a combinatorial family $\mathcal{A}$ is the function defined by the power series

$$g(x) = a_0 + a_1 x + a_2 x^2 + \cdots = \sum_{n=0}^{\infty} a_n x^n$$

where $a_n = |\mathcal{A}_n|$, the number of objects of $\mathcal{A}$ with size n.

Example 2.1

The generating function of the family $\mathcal{B}$ of binary strings defined in Example 1.1 is

$$g(x) = \sum_{n=0}^{\infty} 2^n x^n = \sum_{n=0}^{\infty} (2x)^n = \frac{1}{1-2x}.$$

Example 2.2

The generating function of the family $\mathcal{P}$ of subsets of a finite set S of cardinality m (Example 1.2) is

$$g(x) = \sum_{n=0}^{m} \binom{m}{n} x^n = (1+x)^m.$$

Note that we have not changed the limits of the summation here to make it finite. The binomial coefficient $\binom{m}{n}$ is zero if $n > m$.

As with any other constructive definition in mathematics, this one too can be used in two ways. If we know the values of all a_ns for a family $\mathcal{A}$, then we can plug these into the definition to get the generating function $g(x)$. This may not seem

useful, as we are usually interested in a_n itself, not $g(x)$. However, given a family $\mathcal{A}$ with unknown a_ns, if we somehow manage to find its generating function $g(x)$, then we can look at the coefficient of x^n in the series expansion of $g(x)$ and this coefficient is exactly a_n. But how could we find the generating function when the a_ns are themselves unknown? That is where the *operations* on families play a role. If we manage to figure out how to compute the generating functions of families resulting from these operations, in terms of the generating functions of the operands, clearly we could build up combinatorial families of interest from ones with known generating functions!

Let $\mathcal{A}$ and $\mathcal{B}$ be two combinatorial families, and consider their disjoint union $\mathcal{C} = \mathcal{A} + \mathcal{B}$. Any object of $\mathcal{C}$ having size n is either an object of $\mathcal{A}$ or an object of $\mathcal{B}$ having size n (with no such object being common to both $\mathcal{A}$ and $\mathcal{B}$). Thus, $c_n = a_n + b_n$ (see Exercise 1.6).

Now, what happens in the product $\mathcal{D} = \mathcal{A}\mathcal{B}$? Recall that for every pair of objects α of $\mathcal{A}$ and β of $\mathcal{B}$, we get an object (α, β) of $\mathcal{A}\mathcal{B}$ having size $\text{size}(\alpha) + \text{size}(\beta)$. If this object is to have size exactly n, then it must be that in the family $\mathcal{A}$, $\text{size}(\alpha) = k$ and in the family $\mathcal{B}$, $\text{size}(\beta) = n - k$ (for some k, $0 \leq k \leq n$). Thus, *every* object of $\mathcal{A}$ having size k can be paired with *every* object of $\mathcal{B}$ having size $n - k$ to give an object of $\mathcal{A}\mathcal{B}$ having size n. The number of such pairings is clearly $a_k b_{n-k}$, for each k (since a_k and b_{n-k} are the respective numbers of objects of $\mathcal{A}$ having size k and objects of $\mathcal{B}$ having size $n - k$). The number of objects of $\mathcal{D} = \mathcal{A}\mathcal{B}$ having size exactly n is therefore $d_n = \sum_{k=0}^{n} a_k b_{n-k}$. This tells us how we could compute the generating functions of families obtained by the disjoint union and product operations.

Theorem 2.1

Let $\mathcal{A}$ and $\mathcal{B}$ be two combinatorial families with generating functions $g_A(x)$ and $g_B(x)$ respectively.

(i) The generating function of the disjoint union $\mathcal{A} + \mathcal{B}$ is given by

$$g_{A+B}(x) = g_A(x) + g_B(x).$$

(ii) The generating function of the product $\mathcal{A}\mathcal{B}$ is given by

$$g_{AB}(x) = g_A(x) g_B(x).$$

To see why (ii) is true, observe that

$$\begin{aligned}\left(\sum_{n=0}^{\infty} a_n x^n\right)\left(\sum_{n=0}^{\infty} b_n x^n\right) &= \left(a_0 + a_1 x + a_2 x^2 + \cdots\right)\left(b_0 + b_1 x + b_2 x^2 + \cdots\right) \\ &= a_0 b_0 + (a_0 b_1 + a_1 b_0)x + (a_0 b_2 + a_1 b_1 + a_2 b_0)x^2 + \\ &\quad (a_0 b_3 + a_1 b_2 + a_2 b_1 + a_3 b_0)x^3 + \cdots \\ &= \sum_{n=0}^{\infty}\left(\sum_{k=0}^{n} a_k b_{n-k}\right)x^n.\end{aligned}$$

We shall now apply these results to our earlier examples. As a first example, let us see whether Theorem 2.1 really works.

Example 2.3

If S and T are two disjoint finite sets with cardinalities s and t respectively, and $\mathcal{P}$ and $\mathcal{Q}$ the families defined by their power sets (Example 1.3), then the product family $R = \mathcal{P}\mathcal{Q}$ is (equivalent to) the family of all subsets of the set $S \cup T$ (whose cardinality is $s + t$). The generating functions of $\mathcal{P}$, $\mathcal{Q}$, and $\mathcal{R}$ are

$$\begin{aligned} g_P(x) &= (1+x)^s \\ g_Q(x) &= (1+x)^t \\ g_R(x) &= (1+x)^{s+t}\end{aligned}$$

respectively (Example 2.2). So $g_R(x) = g_P(x)g_Q(x)$ as expected.

Recall from Exercises 1.4 and 1.5 that if $\mathcal{B}$, $\mathcal{E}$, and $\mathcal{O}$ respectively denote the families of all binary strings, the strings with an even number of 1s, and the strings with an odd number of 1s, then $\mathcal{B} = \mathcal{E} + \mathcal{O}$ and $\mathcal{O} = \mathcal{E}\mathcal{T} + \mathcal{O}\mathcal{F}$ (where $\mathcal{T}$ is the family containing only the string 1, and $\mathcal{F}$ is the family containing only the string 0). The former equation is justified by observing that every string with an odd number of 1s (an object of $\mathcal{O}$) is obtained by either appending 1 (the object of $\mathcal{T}$) to a string in $\mathcal{E}$ or appending 0 (the object of $\mathcal{F}$) to a string in $\mathcal{O}$.

Example 2.4

Let $g(x)$ and $h(x)$ be the generating functions of $\mathcal{E}$, and $\mathcal{O}$ respectively. Note that the generating functions of both the families $\mathcal{T}$ and $\mathcal{F}$ are $g_T(x) = g_F(x) = x$. Since $\mathcal{E} + \mathcal{O} = \mathcal{B}$ and $\mathcal{O} = \mathcal{E}\mathcal{T} + \mathcal{O}\mathcal{F}$, application of Theorem 2.1 gives us

$$g(x) + h(x) = \frac{1}{1-2x} \quad \text{(generating function of } \mathcal{B}\text{)}$$

$$h(x) = g(x)x + h(x)x = (g(x) + h(x))x = \frac{x}{1-2x}$$

and therefore

$$g(x) = \frac{1-x}{1-2x} = 1 + \frac{x}{1-2x}.$$

Now to find e_n and o_n, we expand $g(x)$ and $h(x)$ in powers of x to get

$$g(x) = 1 + \sum_{n=1}^{\infty} 2^{n-1}x^n \qquad h(x) = \sum_{n=1}^{\infty} 2^{n-1}x^n.$$

Thus, $e_0 = 1$, $o_0 = 0$ and $e_n = o_n = 2^{n-1}$ for $n > 0$.