Chapter 11

Counting

Counting is the process of creating a bijection between a set we want to count and some set whose size we already know. Typically this second set will be a finite ordinal $[n] = \{0, 1, \ldots, n-1\}$.[1]

Counting a set A using a bijection $f : A \to [n]$ gives its size $|A| = n$; this size is called the **cardinality** of n. As a side effect, it also gives a well-ordering of A, since $[n]$ is well-ordered as we can define $x \leq y$ for x, y in A by $x \leq y$ if and only if $f(x) \leq f(y)$. Often the quickest way to find f is to line up all the elements of A in a well-ordering and then count them off: the smallest element of A gets mapped to 0, the next smallest to 1, and so on. Stripped of the mathematical jargon, this is exactly what you were taught to do as a small child.

Usually we will not provide an explicit bijection to compute the size of a set, but instead will rely on standard counting principles based on how we constructed the set. The branch of mathematics that studies sets constructed by combining other sets is called **combinatorics**, and the sub-branch that counts these sets is called **enumerative combinatorics**. In this chapter, we're going to give an introduction to enumerative combinatorics, but this basically just means counting.

For infinite sets, cardinality is a little more complicated. The basic idea is that we define $|A| = |B|$ if there is a bijection between them. This gives an equivalence relation on sets[2], and we define $|A|$ to be the equivalence class of this equivalence relation that contains A. For the finite case we represent

[1] Starting from 0 is traditional in computer science, because it makes indexing easier. Normal people count to n using $\{1, 2, \ldots, n\}$.

[2] Reflexivity: the identity function is a bijection from A to A. Symmetry: if $f : A \to B$ is a bijection, so is $f^{-1} : B \to A$. Transitivity: if $f : A \to B$ and $g : B \to C$ are bijections, so is $(g \circ f) : A \to C$.

the equivalence classes by taking representative elements $[n]$.

For the most part we will concentrate on counting finite sets, but will mention where the rules for finite sets break down with infinite sets.

11.1 Basic counting techniques

Our goal here is to compute the size of some set of objects, e.g., the number of subsets of a set of size n, the number of ways to put k cats into n boxes so that no box gets more than one cat, etc.

In rare cases we can use the definition of the size of a set directly, by constructing a bijection between the set we care about and some canonical set $[n]$. For example, the set $S_n = \{x \in \mathbb{N} \mid x < n^2 \wedge \exists y : x = y^2\}$ has exactly n members, because we can generate it by applying the one-to-one correspondence $f(y) = y^2$ to the set $\{0, 1, 2, 3, \ldots, n-1\} = [n]$. But most of the time constructing an explicit one-to-one correspondence is too time-consuming or too hard, so instead we will show how to map set-theoretic operations to arithmetic operations, so that from a set-theoretic construction of a set we can often directly read off an arithmetic computation that gives the size of the set.

11.1.1 Equality: reducing to a previously-solved case

If we can produce a bijection between a set A whose size we don't know and a set B whose size we do, then we get $|A| = |B|$. Pretty much all of our proofs of cardinality will end up looking like this.

11.1.2 Inequalities: showing $|A| \leq |B|$ and $|B| \leq |A|$

We write $|A| \leq |B|$ if there is an injection $f : A \to B$, and similarly $|B| \leq |A|$ if there is an injection $g : B \to A$. If both conditions hold, then there is a bijection between A and B, showing $|A| = |B|$. This fact is trivial for finite sets, but for infinite sets—even though it is still true—the actual construction of the bijection is a little trickier.[3]

[3]The claim for general sets is known as the Cantor-Bernstein-Schroeder theorem. One way to prove this is to assume that A and B are disjoint and construct a (not necessarily finite) graph whose vertex set is $A \cup B$ and that has edges for all pairs $(a, f(a))$ and $(b, g(b))$. It can then be shown that the connected components of this graph consist of (1) finite cycles, (2) doubly-infinite paths (i.e., paths with no endpoint in either direction), (3) infinite paths with an initial vertex in A, and (4) infinite paths with an initial vertex in B. For vertexes in all but the last class of components, define $h(x)$ to be $f(x)$ if x is in A and $f^{-1}(x)$ if x is in B. (Note that we are abusing notation slightly here by defining $f^{-1}(x)$

Similarly, if we write $|A| \geq |B|$ to indicate that there is a surjection from A to B, then $|A| \geq |B|$ and $|B| \geq |A|$ implies $|A| = |B|$. The easiest way to show this is to observe that if there is a surjection $f : A \to B$, then we can get an injection $f' : B \to A$ by letting $f'(y)$ be any element of $\{x \mid f(x) = y\}$, thus reducing to the previous case (this requires the Axiom of Choice, but pretty much everybody assumes the Axiom of Choice). Showing an injection $f : A \to B$ and a surjection $g : A \to B$ also works.

For example, $|\mathbb{Q}| = |\mathbb{N}|$. Proof: $|\mathbb{N}| \leq |\mathbb{Q}|$ because we can map any n in $\mathbb{N}$ to the same value in $\mathbb{Q}$; this is clearly an injection. To show $|\mathbb{Q}| \leq |\mathbb{N}|$, observe that we can encode any element $\pm p/q$ of $\mathbb{Q}$, where p and q are both natural numbers, as a triple (s, p, q) where ($s \in \{0, 1\}$ indicates $+$ (0) or $-$ (1); this encoding is clearly injective. Then use the Cantor pairing function (§3.7.1) twice to crunch this triple down to a single natural number, getting an injection from $\mathbb{Q}$ to $\mathbb{N}$.

11.1.3 Addition: the sum rule

The **sum rule** computes the size of $A \cup B$ when A and B are disjoint.

Theorem 11.1.1. *If A and B are finite sets with $A \cap B = \emptyset$, then*

$$|A \cup B| = |A| + |B|.$$

Proof. Let $f : A \to [|A|]$ and $g : B \to [|B|]$ be bijections. Define $h : A \cup B \to [|A| + |B|]$ by the rule $h(x) = f(x)$ for $x \in A$, $h(x) = |A| + g(x)$ for $x \in B$.

To show that this is a bijection, define $h^{-1}(y)$ for y in $[|A| + |b|]$ to be $f^{-1}(y)$ if $y < |A|$ and $g^{-1}(y - |A|)$ otherwise. Then for any y in $[|A| + |B|]$, either

1. $0 \leq y < |A|$, y is in the codomain of f (so $h^{-1}(y) = f^{-1}(y) \in A$ is well-defined), and $h(h^{-1}(y)) = f(f^{-1}(y)) = y$.

2. $|A| \leq y < |A| + |B|$. In this case $0 \leq y - |A| < |B|$, putting $y - |A|$ in the codomain of g and giving $h(h^{-1}(y)) = g(g^{-1}(y - |A|)) + |A| = y$.

So h^{-1} is in fact an inverse of h, meaning that h is a bijection. □

---

to be the unique y that maps to x when it exists.) For the last class of components, the initial B vertex is not the image of any x under f; so for these we define $h(x)$ to be $g(x)$ if x is in B and $g^{-1}(x)$ if x is in A. This gives the desired bijection h between A and B.

In the case where A and B are not disjoint, we can make them disjoint by replacing them with $A' = \{0\} \times A$ and $B' = \{1\} \times B$. (This is a pretty common trick for enforcing disjoint unions.)

One way to think about this proof is that we are constructing a total order on $A \cup B$ by putting all the A elements before all the B elements. This gives a straightforward bijection with $[|A| + |B|]$ by the usual preschool trick of counting things off in order.

Generalizations: If $A_1, A_2, A_3 \ldots A_k$ are **pairwise disjoint** (i.e., $A_i \cap A_j = \emptyset$ for all $i \neq j$), then

$$\left| \bigcup_{i=1}^{k} A_i \right| = \sum_{i=1}^{k} |A_i|.$$

The proof is by induction on k.

Example: As I was going to Saint Ives, I met a man with 7 wives, 28 children, 56 grandchildren, and 122 great-grandchildren. Assuming these sets do not overlap, how many people did I meet? Answer: 1+7+28+56+122=214.

11.1.3.1 For infinite sets

The sum rule works for infinite sets, too; technically, the sum rule is used to *define* $|A| + |B|$ as $|A \cup B|$ when A and B are disjoint. This makes cardinal arithmetic a bit wonky: if at least one of A and B is infinite, then $|A| + |B| = \max(|A|, |B|)$, since we can space out the elements of the larger of A and B and shove the elements of the other into the gaps.

11.1.3.2 The Pigeonhole Principle

A consequence of the sum rule is that if A and B are both finite and $|A| > |B|$, you can't have an injection from A to B. The proof is by contraposition. Suppose $f : A \to B$ is an injection. Write A as the union of $f^{-1}(x)$ for each $x \in B$, where $f^{-1}(x)$ is the set of y in A that map to x. Because each $f^{-1}(x)$ is disjoint, the sum rule applies; but because f is an injection there is at most one element in each $f^{-1}(x)$. It follows that $|A| = \sum_{x \in B} |f^{-1}(x)| \leq \sum_{x \in B} 1 = |B|$. (Question: Why doesn't this work for infinite sets?)

The Pigeonhole Principle generalizes in an obvious way to functions with larger domains; if $f : A \to B$, then there is some x in B such that $|f^{-1}(x)| \geq |A|/|B|$.

11.1.4 Subtraction

For any sets A and B, A is the disjoint union of $A \cap B$ and $A \setminus B$. So $|A| = |A \cap B| + |A \setminus B|$ (for finite sets) by the sum rule. Rearranging gives

$$|A \setminus B| = |A| - |A \cap B|. \tag{11.1.1}$$

What makes (11.1.1) particularly useful is that we can use it to compute the size of $A \cup B$ even if A and B overlap. The intuition is that if we just add $|A|$ and $|B|$, then we count every element of $A \cap B$ twice; by subtracting off $|A \cap B|$ we eliminate the overcount. Formally, we have

Theorem 11.1.2. *For any finite sets A and B,*

$$|A \cup B| = |A| + |B| - |A \cap B|.$$

Proof. Compute

$$\begin{aligned} |A \cup B| &= |A \cap B| + |A \setminus B| + |B \setminus A| \\ &= |A \cap B| + (|A| - |A \cap B|) + (|B| - |A \cap B|) \\ &= |A| + |B| - |A \cap B|. \end{aligned}$$

□

This is a special case of the **inclusion-exclusion formula**, which can be used to compute the size of the union of many sets using the size of pairwise, triple-wise, etc. intersections of the sets. See §11.2.4 for the general rule.

11.1.4.1 Inclusion-exclusion for infinite sets

Subtraction doesn't work very well for infinite quantities (while $\aleph_0 + \aleph_0 = \aleph_0$, that doesn't mean $\aleph_0 = 0$). So the closest we can get to the inclusion-exclusion formula is that $|A| + |B| = |A \cup B| + |A \cap B|$. If at least one of A or B is infinite, then $|A \cup B|$ is also infinite, and since $|A \cap B| \leq |A \cup B|$ we have $|A \cup B| + |A \cap B| = |A \cup B|$ by the bizarre rules of cardinal arithmetic. So for infinite sets we have the rather odd result that $|A \cup B| = |A| + |B| = \max(|A|, |B|)$ whether the sets overlap or not.

11.1.4.2 Combinatorial proof

We can prove $|A| + |B| = |A \cup B| + |A \cap B|$ combinatorially, by turning both sides of the equation into disjoint unions (so the sum rule works) and then providing an explicit bijection between the resulting sets. The trick is that we can always force a union to be disjoint by tagging the elements with extra information; so on the left-hand side we construct $L = \{0\} \times A \cup \{1\} \times B$, and on the right-hand side we construct $R = \{0\} \times (A \cup B) \cup \{1\} \times (A \cap B)$. It is easy to see that both unions are disjoint, because we are always taking the union of a set of ordered pairs that start with 0 with a set of ordered pairs that start with 1, and no ordered pair can start with both tags; it follows that $|L| = |A| + |B|$ and $|R| = |A \cup B| + |A \cap B|$. Now define the function $f : L \to R$ by the rule

$$f((0, x)) = (0, x).$$
$$f((1, x)) = (1, x) \text{if } x \in B \cap A.$$
$$f((1, x)) = (0, x) \text{if } x \in B \setminus A.$$

Observe that f is surjective, because for any $(0, x)$ in $\{0\} \times (A \cup B)$, either x is in A and $(0, x) = f((0, x))$ where $(0, x) \in L$, or x is in $B \setminus A$ and $(0, x) = f((1, x))$ where $(1, x) \in L$. It is also true that f is injective; the only way for it not to be is if $f((0, x)) = f((1, x)) = (0, x)$ for some x. Suppose this occurs. Then $x \in A$ (because of the 0 tag) and $x \in B \setminus A$ (because $(1, x)$ is only mapped to $(0, x)$ if $x \in B \setminus A$). But x can't be in both A and $B \setminus A$, so we get a contradiction.

11.1.5 Multiplication: the product rule

The **product rule** says that Cartesian product maps to arithmetic product. Intuitively, we line the elements (a, b) of $A \times B$ in lexicographic order and count them off. This looks very much like packing a two-dimensional array in a one-dimensional array by mapping each pair of indices (i, j) to $i \cdot |B| + j$.

Theorem 11.1.3. *For any finite sets A and B,*

$$|A \times B| = |A| \cdot |B|.$$

Proof. The trick is to order $A \times B$ lexicographically and then count off the elements. Given bijections $f : A \to [|A|]$ and $g : B \to [|B|]$, define $h : (A \times B) \to [|A| \cdot |B|]$ by the rule $h((a, b)) = a \cdot |B| + b$. The division

algorithm recovers a and b from $h(a, b)$ by recovering the unique natural numbers q and r such that $h(a, b) = q \cdot |B| + r$ and $0 \leq b < |B|$ and letting $a = f^{-1}(q)$ and $b = g^{-1}(r)$. □

The general form is

$$\left| \prod_{i=1}^{k} A_i \right| = \prod_{i=1}^{k} |A_i|,$$

where the product on the left is a Cartesian product and the product on the right is an ordinary integer product.

11.1.5.1 Examples

- As I was going to Saint Ives, I met a man with seven sacks, and every sack had seven cats. How many cats total? Answer: Label the sacks $0, 1, 2, \ldots, 6$, and label the cats in each sack $0, 1, 2, \ldots, 6$. Then each cat can be specified uniquely by giving a pair (sack number, cat number), giving a bijection between the set of cats and the set 7×7. Since $|7 \times 7| = 7 \cdot 7 = 49$, we have 49 cats.

- Dr. Frankenstein's trusty assistant Igor has brought him 6 torsos, 4 brains, 8 pairs of matching arms, and 4 pairs of legs. How many different monsters can Dr Frankenstein build? Answer: there is a one-to-one correspondence between possible monsters and 4-tuples of the form (torso, brain, pair of arms, pair of legs); the set of such 4-tuples has $6 \cdot 4 \cdot 8 \cdot 4 = 728$ members.

- How many different ways can you order n items? Call this quantity $n!$ (pronounced "n **factorial**"). With 0 or 1 items, there is only one way; so we have $0! = 1! = 1$. For $n > 1$, there are n choices for the first item, leaving $n - 1$ items to be ordered. From the product rule we thus have $n! = n \cdot (n - 1)!$, which we can expand out as $\prod_{i=1}^{n} i$, our previous definition of $n!$.

11.1.5.2 For infinite sets

The product rule also works for infinite sets, because we again use it as a definition: for any A and B, $|A| \cdot |B|$ is defined to be $|A \times B|$. One oddity for infinite sets is that this definition gives $|A| \cdot |B| = |A| + |B| = \max(|A|, |B|)$, because if at least one of A and B is infinite, it is possible to construct a bijection between $A \times B$ and the larger of A and B. Infinite sets are strange.

11.1.6 Exponentiation: the exponent rule

Given sets A and B, let A^B be the set of functions $f : B \to A$. Then $\left|A^B\right| = |A|^{|B|}$.

If $|B|$ is finite, this is just a $|B|$-fold application of the product rule: we can write any function $f : B \to A$ as a sequence of length $|B|$ that gives the value in A for each input in B. Since each element of the sequence contributes $|A|$ possible choices, we get $|A|^{|B|}$ choices total.

For infinite sets, the exponent rule is a definition of $|A|^{|B|}$. Some simple facts are that $n^\alpha = 2^\alpha$ whenever n is finite and α is infinite (this comes down to the fact that we can represent any element of $[n]$ as a finite sequence of bits) and $\alpha^n = \alpha$ under the same conditions (follows by induction on n from $\alpha \cdot \alpha = \alpha$). When α and β are both infinite, many strange things can happen.

To give a flavor of how exponentiation works for arbitrary sets, here's a combinatorial proof of the usual arithmetic fact that $x^a x^b = x^{a+b}$, for any cardinal numbers x, a, and b. Let $x = |X|$ and let $a = |A|$ and $b = |B|$ where A and B are disjoint (we can always use the tagging trick that we used for inclusion-exclusion to make A and B be disjoint). Then $x^a x^b = \left|X^A \times X^B\right|$ and $x^{a+b} = \left|X^{A \cup B}\right|$. We will now construct an explicit bijection $f : X^{A \cup B} \to X^A \times X^B$. The input to f is a function $g : A \cup B \to X$; the output is a pair of functions $(g_A : A \to X, g_B : B \to X)$. We define g_A by $g_A(x) = g(x)$ for all x in A (this makes g_A the **restriction** of g to A, usually written as $g \restriction A$ or $g|A$); similarly $g_B = g \restriction B$. This is easily seen to be a bijection; if $g = h$, then $f(g) = (g \restriction A, g \restriction B) = f(h) = (h \restriction A, h \restriction B)$, and if $g \neq h$ there is some x for which $g(x) \neq h(x)$, implying $g \restriction A \neq h \restriction A$ (if x is in A) or $g \restriction B \neq h \restriction B$ (if x is in B).

11.1.6.1 Counting injections

Counting injections from a k-element set to an n-element set corresponds to counting the number of ways $P(n, k)$ we can pick an ordered subset of k of n items without replacement, also known as picking a **k-permutation**. (The k elements of the domain correspond to the k positions in the order.)

There are n ways to pick the first item, $n - 1$ to pick the second, and so forth, giving a total of

$$P(n, k) = \prod_{i=n-k+1}^{n} i = \frac{n!}{(n-k)!}$$

such k-permutations by the product rule.

Among combinatorialists, the notation $(n)_k$ (pronounced "n **lower-factorial** k") is more common than $P(n,k)$ for $n\cdot(n-1)\cdot(n-2)\cdot\ldots\cdot(n-k+1)$. As an extreme case we have $(n)_n = n\cdot(n-1)\cdot(n-2)\cdot\ldots\cdot(n-n+1) = n\cdot(n-1)\cdot(n-2)\cdot\ldots\cdot 1 = n!$, so $n!$ counts the number of **permutations** of n.

This gives us three tools for counting functions between sets: n^k counts the number of functions from a k-element set to an n-element set, $(n)_k$ counts the number of injections from a k-element set to an n-element set, and $n!$ counts the number of bijections between two n-element sets (or from an n-element set to itself).

Counting surjections is messier. If you really need to do this, you will need to use **Stirling numbers**; see [GKP94, Chapter 6] or [Sta97, p. 33].

11.1.7 Division: counting the same thing in two different ways

An old farm joke:

Q: How do you count a herd of cattle?

A: Count the legs and divide by four.

Sometimes we can compute the size of a set S by using it (as an unknown variable) to compute the size of another set T (as a function of $|S|$), and then using some other way to count T to find its size, finally solving for $|S|$. This is known as **counting two ways** and is surprisingly useful when it works. We will assume that all the sets we are dealing with are finite, so we can expect things like subtraction and division to work properly.

11.1.7.1 Binomial coefficients

What is $|S_k|$? Answer: First we'll count the number m of sequences of k elements of S with no repetitions. We can get such a sequence in two ways:

1. By picking a size-k subset A and then choosing one of $k!$ ways to order the elements. This gives $m = |S_k|\cdot k!$.

2. By choosing the first element in one of n ways, the second in one of $n-1$, the third in one of $n-2$ ways, and so on until the k-th element, which can be chosen in one of $n-k+1$ ways. This gives $m = (n)_k = n\cdot(n-1)\cdot(n-2)\cdot\ldots(n-k+1)$, which can be written as $n!/(n-k)!$. (Here we are using the factors in $(n-k)!$ to cancel out the factors in $n!$ that we don't want.)

So we have $m = |S_k| \cdot k! = n!/(n-k)!$, from which we get

$$|S_k| = \frac{n!}{k! \cdot (n-k)!}.$$

This quantity turns out to be so useful that it has a special notation:

$$\binom{n}{k} \stackrel{\text{def}}{=} \frac{n!}{k! \cdot (n-k)!}.$$

where the left-hand side is known as a **binomial coefficient** and is pronounced "n choose k." We discuss binomial coefficients at length in §11.2. The secret of why it's called a binomial coefficient will be revealed when we talk about generating functions in §11.3.

11.1.7.2 Multinomial coefficients

Here's a generalization of binomial coefficients: let the **multinomial coefficient**

$$\binom{n}{n_1\ n_2\ \ldots\ n_k}$$

be the number of different ways to distribute n items among k bins where the i-th bin gets exactly n_i of the items and we don't care what order the items appear in each bin. (Obviously this only makes sense if $n_1 + n_2 + \cdots + n_k = n$.) Can we find a simple formula for the multinomial coefficient?

Here are two ways to count the number of permutations of the n-element set:

1. Pick the first element, then the second, etc., to get $n!$ permutations.
2. Generate a permutation in three steps:
   (a) Pick a partition of the n elements into blocks of size $n_1, n_2, \ldots n_k$.
   (b) Order the elements of each block.
   (c) Paste the blocks together into a single ordered list.

There are

$$\binom{n}{n_1\ n_2\ \ldots\ n_k}$$

ways to pick the partition and

$$n_1! \cdot n_2! \cdots n_k!$$

ways to order the elements of all the groups, so we have

$$n! = \binom{n}{n_1\ n_2\ \ldots\ n_k} \cdot n_1! \cdot n_2! \cdots n_k!,$$

which we can solve to get

$$\binom{n}{n_1\ n_2\ \ldots\ n_k} = \frac{n!}{n_1! \cdot n_2! \cdots n_k!}.$$

This also gives another way to derive the formula for a binomial coefficient, since

$$\binom{n}{k} = \binom{n}{k\ (n-k)} = \frac{n!}{k! \cdot (n-k)!}.$$

11.1.8 Applying the rules

If you're given some strange set to count, look at the structure of its description:

- If it's given by a rule of the form x is in S if either $P(x)$ or $Q(x)$ is true, use the sum rule (if P and Q are mutually exclusive) or inclusion-exclusion. This includes sets given by recursive definitions, e.g. x is a tree of depth at most k if it is either (a) a single leaf node (provided $k > 0$) or (b) a root node with two subtrees of depth at most $k - 1$. The two classes are disjoint so we have $T(k) = 1 + T(k-1)^2$ with $T(0) = 0$.[4]

- For objects made out of many small components or resulting from many small decisions, try to reduce the description of the object to something previously known, e.g. (a) a word of length k of letters from an alphabet of size n allowing repetition (there are n^k of them, by the product rule); (b) a word of length k not allowing repetition (there are $(n)_k$ of them—or $n!$ if $n = k$); (c) a subset of k distinct things from a set of size n, where we don't care about the order (there are $\binom{n}{k}$ of them); any subset of a set of n things (there are 2^n of them—this

[4] Of course, just setting up a recurrence doesn't mean it's going to be easy to actually solve it.

is a special case of (a), where the alphabet encodes non-membership as 0 and membership as 1, and the position in the word specifies the element). Some examples:

  - The number of games of Tic-Tac-Toe assuming both players keep playing until the board is filled is obtained by observing that each such game can be specified by listing which of the 9 squares are filled in order, giving $9! = 362880$ distinct games. Note that we don't have to worry about which of the 9 moves are made by X and which by O, since the rules of the game enforce it. (If we only consider games that end when one player wins, this doesn't work: probably the easiest way to count such games is to send a computer off to generate all of them. This gives 255168 possible games and 958 distinct final positions.)
  - The number of completely-filled-in Tic-Tac-Toe boards can be obtained by observing that any such board has 5 X's and 4 O's. So there are $\binom{9}{5} = 126$ such positions. (Question: Why would this be smaller than the actual number of final positions?)

Sometimes reducing to a previous case requires creativity. For example, suppose you win n identical cars on a game show and want to divide them among your k greedy relatives. Assuming that you don't care about fairness, how many ways are there to do this?

- If it's OK if some people don't get a car at all, then you can imagine putting n cars and $k-1$ dividers in a line, where relative 1 gets all the cars up to the first divider, relative 2 gets all the cars between the first and second dividers, and so forth up to relative k who gets all the cars after the $(k-1)$-th divider. Assume that each car—and each divider—takes one parking space. Then you have $n+k-1$ parking spaces with $k-1$ dividers in them (and cars in the rest). There are exactly $\binom{n+k-1}{k-1}$ ways to do this.

- Alternatively, suppose each relative demands at least 1 car. Then you can just hand out one car to each relative to start with, leaving $n-k$ cars to divide as in the previous case. There are $\binom{(n-k)+k-1}{k-1} = \binom{n-1}{k-1}$ ways to do this.

  As always, whenever some counting problem turns out to have an easier answer than expected, it's worth trying to figure out if there is a more direct combinatorial proof. In this case we want to encode assignments

of at least one of n cars to k people, so that this corresponds to picking $k-1$ out of $n-1$ things. One way to do this is to imagine lining up all n cars, putting each relative in front of one of the cars, and giving them that car plus any car to the right until we hit the next relative. In order for this to assign all the cars, we have to put the leftmost relative in front of the leftmost car. This leaves $n-1$ places for the $k-1$ remaining relatives, giving $\binom{n-1}{k-1}$ choices.

Finding correspondences like this is a central part of enumerative combinatorics, the branch of mathematics that deals with counting things.

11.1.9 An elaborate counting problem

Suppose you have the numbers $\{1, 2, \ldots, 2n\}$, and you want to count how many sequences of k of these numbers you can have that are (a) increasing ($a[i] < a[i+1]$ for all i), (b) decreasing ($a[i] \geq a[i+1]$ for all i), or (c) made up only of even numbers.

This is the union of three sets A, B, and C, corresponding to the three cases. The first step is to count each set individually; then we can start thinking about applying inclusion-exclusion to get the size of the union.

For A, any increasing sequence can be specified by choosing its elements (the order is determined by the assumption it is increasing). So we have $|A| = \binom{2n}{k}$.

For B, by symmetry we have $|B| = |A| = \binom{2n}{k}$.

For C, we are just looking at n^k possible sequences, since there are n even numbers we can put in each position.

Inclusion-exclusion says that $|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cup B \cup C|$. It's not hard to see that $A \cap B = \emptyset$ when k is at least 2,[5] so we can reduce this to $|A| + |B| + |C| - |A \cap C| - |B \cap C|$. To count $A \cap C$, observe that we are now looking at increasing sequences chosen from the n possible even numbers; so there are exactly $\binom{n}{k}$ of them, and similarly for $B \cap C$. Summing up gives a total of

$$\binom{2n}{k} + \binom{2n}{k} + n^k - \binom{n}{k} - \binom{n}{k} = 2\left(\binom{2n}{k} - \binom{n}{k}\right) + n^k$$

sequences satisfying at least one of the criteria.

[5] It's even easier to assume that $A \cap B = \emptyset$ always, but for $k = 1$ any sequence is both increasing and nonincreasing, since there are no pairs of adjacent elements in a 1-element sequence to violate the property.

Note that we had to assume $k = 2$ to get $A \cap B = \emptyset$, so this formula might require some adjustment for $k < 2$. In fact we can observe immediately that the unique empty sequence for $k = 1$ fits in all of A, B, and C, so in this case we get 1 winning sequence (which happens to be equal to the value in the formula, because here $A \cap B = \emptyset$ for other reasons), and for $k = 1$ we get $2n$ winning sequences (which is less than the value $3n$ given by the formula).

To test that the formula works for at least some larger values, let $n = 3$ and $k = 2$. Then the formula predicts $2\left(\binom{6}{2} - \binom{3}{2}\right) + 3^2 = 2(15 - 3) + 9 = 33$

total sequences.[6] And here they are:

$$
\begin{array}{c}
(1,2)\\
(1,3)\\
(1,4)\\
(1,5)\\
(1,6)\\
(2,1)\\
(2,2)\\
(2,3)\\
(2,4)\\
(2,5)\\
(2,6)\\
(3,1)\\
(3,2)\\
(3,4)\\
(3,5)\\
(3,6)\\
(4,1)\\
(4,2)\\
(4,3)\\
(4,4)\\
(4,5)\\
(4,6)\\
(5,1)\\
(5,2)\\
(5,3)\\
(5,4)\\
(5,6)\\
(6,1)\\
(6,2)\\
(6,3)\\
(6,4)\\
(6,5)\\
(6,6)
\end{array}
$$

[6] Without looking at the list, can you say which 3 of the $6^2 = 36$ possible length-2 sequences are missing?

11.1.10 Further reading

Rosen [Ros12] does basic counting in Chapter 6 and more advanced counting (including solving recurrences and using generating functions) in chapter 8. Biggs [Big02] gives a basic introduction to counting in Chapters 6 and 10, with more esoteric topics in Chapters 11 and 12. Graham *et al.* [GKP94] have quite a bit on counting various things.

Combinatorics largely focuses on counting rather than efficient algorithms for constructing particular combinatorial objects. The book *Constructive Combinatorics*, by Stanton and White, [SW86] remedies this omission, and includes algorithms not only for enumerating all instances of various classes of combinatorial objects but also for finding the i-th such instance in an appropriate ordering without having to generate all previous instances (**unranking**) and the inverse operation of finding the position of a particular object in an appropriate ordering (**ranking**).

11.2 Binomial coefficients

The **binomial coefficient** "n choose k", written

$$\binom{n}{k} = \frac{(n)_k}{k!} = \frac{n!}{k! \cdot (n-k)!}, \tag{11.2.1}$$

counts the number of k-element subsets of an n-element set. (See §11.1.7.1 for how to derive (11.2.1).)

The name arises from the **binomial theorem**, which in the following form was first proved by Isaac Newton:

Theorem 11.2.1 (Binomial theorem)**.** *For any $n \in \mathbb{R}$,*

$$(x+y)^n = \sum_{k=0}^{\infty} \binom{n}{k} x^k y^{n-k}, \tag{11.2.2}$$

provided the sum converges.

A sufficient condition for the sum converging is $|x/y| < 1$. For the general version of the theorem, $\binom{n}{k}$ is defined as $(n)_k / k!$, which works even if n is not a non-negative integer. The usual proof requires calculus.

In the common case when n is a non-negative integer, we can limit ourselves to letting k range from 0 to n. The reason is that $\binom{n}{k} = 0$ when n

is a non-negative integer and $k > n$. This gives the more familiar version

$$(x+y)^n = \sum_{k=0}^{n} \binom{n}{k} x^k y^{n-k}. \tag{11.2.3}$$

The connection between (11.2.3) and counting subsets is straightforward: expanding $(x+y)^n$ using the distributive law gives 2^n terms, each of which is a unique sequence of n x's and y's. If we think of the x's in each term as labeling a subset of the n positions in the term, the terms that get added together to get $x^k y^{n-k}$ correspond one-to-one to subsets of size k. So there are $\binom{n}{k}$ such terms, accounting for the coefficient on the right-hand side.

11.2.1 Recursive definition

If we don't like computing factorials, we can also compute binomial coefficients recursively. This may actually be less efficient for large n (we need to do $\Theta(n^2)$ additions instead of $\Theta(n)$ multiplications and divisions), but the recurrence gives some insight into the structure of binomial coefficients.

Base cases:

- If $k = 0$, then there is exactly one zero-element set of our n-element set—it's the empty set—and we have $\binom{n}{0} = 1$.
- If $k > n$, then there are no k-element subsets, and we have $\forall k > n : \binom{n}{k} = 0$.

Recursive step: We'll use **Pascal's identity**, which says that

$$\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}.$$

The easiest proof of this identity is combinatorial, which means that we will construct an explicit bijection between a set counted by the left-hand side and a set counted by the right-hand side. This is often one of the best ways of understanding simple binomial coefficient identities.

On the left-hand side, we are counting all the k-element subsets of an n-element set S. On the right hand side, we are counting two different collections of sets: the $(k-1)$-element and k-element subsets of an $(n-1)$-element set. The trick is to recognize that we get an $(n-1)$-element set S' from our original set by removing one of the elements x. When we do this, we affect the subsets in one of two ways:

1. If the subset doesn't contain x, it doesn't change. So there is a one-to-one correspondence (the identity function) between k-subsets of S that don't contain x and k-subsets of S'. This bijection accounts for the first term on the right-hand side.

2. If the subset does contain x, then we get a $(k-1)$-element subset of S' when we remove it. Since we can go back the other way by reinserting x, we get a bijection between k-subsets of S that contain x and $(k-1)$-subsets of S'. This bijection accounts for the second term on the right-hand side.

Adding the two cases together (using the sum rule), we conclude that the identity holds.

Using the base case and Pascal's identity, we can construct **Pascal's triangle**, a table of values of binomial coefficients:

$$\begin{array}{cccccc} 1 & & & & & \\ 1 & 1 & & & & \\ 1 & 2 & 1 & & & \\ 1 & 3 & 3 & 1 & & \\ 1 & 4 & 6 & 4 & 1 & \\ 1 & 5 & 10 & 10 & 5 & 1 \\ \ldots & & & & & \end{array}$$

Each row corresponds to increasing values of n, and each column to increasing values of k, with $\binom{0}{0}$ in the upper left-hand corner. To compute each entry, we add together the entry directly above it and the entry diagonally above and to the left.

11.2.1.1 Pascal's identity: algebraic proof

Using the binomial theorem plus a little bit of algebra, we can prove Pascal's identity without using a combinatorial argument (this is not necessarily an improvement). The additional fact we need is that if we have two equal series

$$\sum_{k=0}^{\infty} a_k x^k = \sum_{k=0}^{\infty} b_k x^k$$

then $a_k = b_k$ for all k.[7]

[7] This is a theorem in analysis if the series represents converges in some open interval around 0, and follows from the ability to extract coefficients from $f(x) = \sum_{k=0}^{\infty} a_k x^k$ by

Here's the proof of Pascal's identity:

$$\begin{aligned}
\sum_{k=0}^{n} \binom{n}{k} x^k &= (1+x)^n \\
&= (1+x)(1+x)^{n-1} \\
&= (1+x)^{n-1} + x(1+x)^{n-1} \\
&= \sum_{k=0}^{n-1} \binom{n-1}{k} x^k + x \sum_{k=0}^{n-1} \binom{n-1}{k} x^k \\
&= \sum_{k=0}^{n-1} \binom{n-1}{k} x^k + \sum_{k=0}^{n-1} \binom{n-1}{k} x^{k+1} \\
&= \sum_{k=0}^{n-1} \binom{n-1}{k} x^k + \sum_{k=1}^{n} \binom{n-1}{k-1} x^k \\
&= \sum_{k=0}^{n} \binom{n-1}{k} x^k + \sum_{k=0}^{n} \binom{n-1}{k-1} x^k \\
&= \sum_{k=0}^{n} \left(\binom{n-1}{k} + \binom{n-1}{k-1} \right) x^k.
\end{aligned}$$

and now we equate matching coefficients to get

$$\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$$

as advertised.

11.2.2 Vandermonde's identity

Vandermonde's identity says that, provided r does not exceed m or n,

$$\binom{m+n}{r} = \sum_{k=0}^{r} \binom{m}{r-k} \binom{n}{k}.$$

11.2.2.1 Combinatorial proof

To pick r elements of an $m+n$ element set, we have to pick some of them from the first m elements and some from the second n elements. Suppose

taking derivatives: $a_k = \frac{1}{k!} f^{(k)}(0)$, where $f^{(k)} = \frac{d^k}{dx^k} f(x)$. Alternatively, we can treat each series as a **formal power series**, which we think of a s an infinite sequence of coefficients on which we can do the usual arithmetic operations without worrying about convergence.

we choose k elements from the last n; there are $\binom{n}{k}$ different ways to do this, and $\binom{m}{r-k}$ different ways to choose the remaining $r-k$ from the first m. This gives (by the product rule) $\binom{m}{r-k}\binom{n}{k}$ ways to choose r elements from the whole set if we limit ourselves to choosing exactly k from the last n. The identity follow by summing over all possible values of k.

11.2.2.2 Algebraic proof

Here we use the fact that, for any sequences of coefficients $\{a_i\}$ and $\{b_i\}$,

$$\left(\sum_{i=0}^{n} a_i x^i\right)\left(\sum_{i=0}^{m} b_i x^i\right) = \sum_{i=0}^{m+n}\left(\sum_{j=0}^{i} a_j b_{i-j}\right) x^i.$$

So now consider

$$\begin{aligned}\sum_{r=0}^{m+n}\binom{m+n}{r}x^r &= (1+x)^{m+n}\\ &= (1+x)^n(1+x)^m\\ &= \left(\sum_{i=0}^{n}\binom{n}{i}x^i\right)\left(\sum_{j=0}^{m}\binom{m}{j}x^j\right)\\ &= \sum_{r=0}^{m+n}\left(\sum_{k=0}^{r}\binom{n}{k}\binom{m}{r-k}\right)x^r.\end{aligned}$$

and equate terms with matching exponents.

Is this more enlightening than the combinatorial version? It depends on what kind of enlightenment you are looking for. In this case the combinatorial and algebraic arguments are counting essentially the same things in the same way, so it's not clear what if any advantage either has over the other. But in many cases it's easier to construct an algebraic argument than a combinatorial one, in the same way that it's easier to do arithmetic using standard grade-school algorithms than by constructing explicit bijections. On the other hand, a combinatorial argument may let you carry other things you know about some structure besides just its size across the bijection, giving you more insight into the things you are counting. The best course is probably to have both techniques in your toolbox.

11.2.3 Sums of binomial coefficients

What is the sum of all binomial coefficients for a given n? We can show

$$\sum_{k=0}^{n} \binom{n}{k} = 2^n$$

combinatorially, by observing that adding up all subsets of an n-element set of all sizes is the same as counting all subsets. Alternatively, apply the binomial theorem to $(1+1)^n$.

Here's another sum, with alternating sign. This is useful if you want to know how the even-k binomial coefficients compare to the odd-k binomial coefficients.

$$\sum_{k=0}^{n} (-1)^k \binom{n}{k} = 0. \text{(Assuming } n \neq 0.\text{)}$$

Proof: $(1-1)^n = 0^n = 0$ when n is nonzero. (When n *is* zero, the 0^n part still works, since $0^0 = 1 = \binom{0}{0}(-1)^0$.)

By now it should be obvious that

$$\sum_{k=0}^{n} 2^k \binom{n}{k} = 3^n.$$

It's not hard to construct more examples of this phenomenon.

11.2.4 The general inclusion-exclusion formula

We've previously seen that $|A \cup B| = |A| + |B| - |A \cap B|$. The generalization of this fact from two to many sets is called the **inclusion-exclusion** formula and says:

Theorem 11.2.2.

$$\left| \bigcup_{i=1}^{n} A_i \right| = \sum_{S \subseteq \{1 \ldots n\}, S \neq \emptyset} (-1)^{|S|+1} \left| \bigcap_{j \in S} A_j \right|. \tag{11.2.4}$$

This rather horrible expression means that to count the elements in the union of n sets A_1 through A_n, we start by adding up all the individual sets $|A_1| + |A_2| + \ldots |A_n|$, then subtract off the overcount from elements that appear in two sets $-|A_1 \cap A_2| - |A_1 \cap A_3| - \ldots$, then add back the resulting undercount from elements that appear in three sets, and so on.

Why does this work? Consider a single element x that appears in k of the sets. We'll count it as $+1$ in $\binom{k}{1}$ individual sets, as -1 in $\binom{k}{2}$ pairs, $+1$ in $\binom{k}{3}$ triples, and so on, adding up to

$$\sum_{i=1}^{k}(-1)^{k+1}\binom{k}{i} = -\left(\sum_{i=1}^{k}(-1)^{k}\binom{k}{i}\right) = -\left(\sum_{i=0}^{k}(-1)^{k}\binom{k}{i} - 1\right) = -(0-1) = 1.$$

11.2.5 Negative binomial coefficients

Though it doesn't make sense to talk about the number of k-subsets of a (-1)-element set, the binomial coefficient $\binom{n}{k}$ has a meaningful value for negative n, which works in the binomial theorem. We'll use the lower-factorial version of the definition:

$$\binom{-n}{k} = (-n)_k / k! = \left(\prod_{i=-n-k+1}^{-n} i\right) / k!.$$

Note we still demand that $k \in \mathbb{N}$; we are only allowed to do funny things with the upper index n.

So for example:

$$\binom{-1}{k} = (-1)_k / k! = \left(\prod_{i=-1-k+1}^{-1} i\right) / k! = \left(\prod_{i=-k}^{-1} i\right) / \left(\prod_{i=1}^{k} i\right) = (-1)^k.$$

An application of this fact is that

$$\frac{1}{1-z} = (1-z)^{-1} = \sum_{n=0}^{\infty}\binom{-1}{n}1^{-1-n}(-z)^n = \sum_{n=0}^{\infty}(-1)^n(-z)^n = \sum_{n=0}^{\infty} z^n.$$

In computing this sum, we had to be careful which of 1 and $-z$ got the n exponent and which got $-1-n$. If we do it the other way, we get

$$\frac{1}{1-z} = (1-z)^{-1} = \sum_{n=0}^{\infty}\binom{-1}{n}1^{n}(-z)^{-1-n} = -\frac{1}{z}\sum_{n=0}^{\infty}\frac{1}{z^n}$$

This turns out to actually be correct, since applying the geometric series formula turns the last line into

$$-\frac{1}{z}\cdot\frac{1}{1-1/z} = -\frac{1}{z-1} = \frac{1}{1-z},$$

but it's a lot less useful.

What happens for a larger upper index? One way to think about $(-n)_k$ is that we are really computing $(n+k-1)_k$ and then negating all the factors, which is equivalent to multiplying the whole expression by $(-1)^k$. So this gives us the identity

$$\begin{aligned}\binom{-n}{k} &= \frac{(-n)_k}{k!} \\ &= (-1)^k \frac{(n+k-1)_k}{k!} \\ &= (-1)^k \binom{n+k-1}{k}.\end{aligned}$$

So, for example,

$$\begin{aligned}\frac{1}{(1-z)^2} &= (1-z)^{-2} \\ &= \sum_n \binom{-2}{n} 1^{-2-n}(-z)^n \\ &= \sum_n (-1)^n \binom{n+1}{n} (-z)^n \\ &= \sum_n (n+1) z^n.\end{aligned}$$

If you are a fan of calculus,[8] you can also get this result by computing

$$\begin{aligned}\frac{1}{(1-z)^2} &= \frac{d}{dz}\frac{1}{1-z} \\ &= \frac{d}{dz}\sum_{n=0}^{\infty} z^n \\ &= \sum_{n=0}^{\infty} \frac{d}{dz} z^n \\ &= \sum_{n=0}^{\infty} n z^{n-1} \\ &= \sum_{n=0}^{\infty} (n+1) z^n.\end{aligned}$$

These facts will be useful when we look at generating functions in §11.3.

[8] Or just got back from reading Appendix H.

11.2.6 Fractional binomial coefficients

Yes, we can do fractional binomial coefficients, too. Exercise: Find the value of

$$\binom{1/2}{n} = \frac{(1/2)_n}{n!}.$$

Like negative binomial coefficients, these don't have an obvious combinatorial interpretation, but can be handy for computing power series of fractional binomial powers like $\sqrt{1+z} = (1+z)^{1/2}$.

11.2.7 Further reading

Graham *et al.* [GKP94] §5.1–5.3 is an excellent source for information about all sorts of facts about binomial coefficients.

11.3 Generating functions

We've seen that in some cases we can use the binomial theorem to express infinite power series like $\sum_n z^n$ as compact expressions like $\frac{1}{1-z}$. The compact representation is called a **generating function** of the series, and manipulating generating functions can be an efficient tool to keep track of series whose coefficients represent sequences of counts of combinatorial objects of different sizes.

11.3.1 Basics

A generating function represents objects of weight n with z^n, and adds all the objects you have up to get a sum $a_0 z^0 + a_1 z^1 + a_2 z^2 + \ldots$, where each a_n counts the number of different objects of weight n. If you are very lucky (or constructed your set of objects by combining simpler sets of objects in certain straightforward ways) there will be some compact expression that is expands to this horrible sum but is easier to write down. Such compact expressions are called **generating functions**, and manipulating them algebraically gives an alternative to actually knowing how to count (Chapter 11).

11.3.1.1 A simple example

We are given some initial prefixes for words: `qu`, `s`, and `t`; some vowels to put in the middle: `a`, `i`, and `oi`; and some suffixes: `d`, `ff`, and `ck`, and we want to calculate the number of words we can build of each length.

One way is to generate all 27 words[9] and sort them by length:

```
sad sid tad tid
quad quid sack saff sick siff soid tack taff tick tiff toid
quack quaff quick quiff quoid soick soiff toick toiff
quoick quoiff
```

This gives us 4 length-3 words, 12 length-4 words, 9 length-5 words, and 2 length-6 words. This is probably best done using a computer, and becomes expensive if we start looking at much larger lists.

An alternative is to solve the problem by judicious use of algebra. Pretend that each of our letters is actually a variable, and that when we concatenate `qu`, `oi`, and `ck` to make `quoick`, we are really multiplying the variables using our usual notation. Then we can express all 27 words as the product $(\mathtt{qu}+\mathtt{s}+\mathtt{t})(\mathtt{a}+\mathtt{i}+\mathtt{oi})(\mathtt{d}+\mathtt{ff}+\mathtt{ck})$. But we don't care about the exact set of words, we just want to know how many we get of each length.

So now we do the magic trick: we replace every variable we've got with a single variable z. For example, this turns `quoick` into $zzzzzz = z^6$, so we can still find the length of a word by reading off the exponent on z. But we can also do this before we multiply everything out, getting

$$\begin{aligned}(zz+z+z)(z+z+zz)(z+zz+zz) &= (2z+z^2)(2z+z^2)(z+2z^2)\\ &= z^3(2+z)^2(1+2z)\\ &= z^3(4+4z+z^2)(1+2z)\\ &= z^3(4+12z+9z^2+2z^3)\\ &= 4z^3+12z^4+9z^5+2z^6.\end{aligned}$$

We can now read off the number of words of each length directly off the coefficients of this polynomial.

11.3.1.2 Why this works

In general, what we do is replace any object of **weight** 1 with z. If we have an object with weight n, we think of it as n weight-1 objects stuck together, i.e., z^n. Disjoint unions are done using addition as in simple counting: $z+z^2$ represents the choice between a weight-1 object and a weight-2 object (which might have been built out of 2 weight-1 objects), while $12z^4$ represents a

[9] We are using *word* in the combinatorial sense of a finite sequence of letters (possibly even the empty sequence) and not the usual sense of a finite, nonempty sequence of letters that actually make sense.

choice between 12 different weight-4 objects. The trick is that when we multiply two expressions like this, whenever two values z^k and z^l collide, the exponents add to give a new value z^{k+l} representing a new object with total weight $k+l$, and if we have something more complex like $(nz^k)(mz^l)$, then the coefficients multiply to give $(nm)z^{k+l}$ different weight $(k+l)$ objects.

For example, suppose we want to count the number of robots we can build given 5 choices of heads, each of weight 2, and 6 choices of bodies, each of weight 5. We represent the heads by $5z^2$ and the bodies by $6z^5$. When we multiply these expressions together, the coefficients multiply (which we want, by the product rule) and the exponents add: we get $5z^2 \cdot 6z^5 = 30z^7$ or 30 robots of weight 7 each.

The real power comes in when we consider objects of different weights. If we add to our 5 weight-2 robot heads two extra-fancy heads of weight 3, and compensate on the body side with three new lightweight weight-4 bodies, our new expression is $(5z^2 + 2z^3)(3z^4 + 6z^5) = 15z^6 + 36z^7 + 12z^8$, giving a possible 15 weight-6 robots, 36 weight-7 robots, and 12 weight-8 robots. The rules for multiplying polynomials automatically tally up all the different cases for us.

This trick even works for infinitely-long polynomials that represent infinite series (such "polynomials" are called **formal power series**). Even though there might be infinitely many ways to pick three natural numbers, there are only finitely many ways to pick three natural numbers whose sum is 37. By computing an appropriate formal power series and extracting the coefficient from the z^{37} term, we can figure out exactly how many ways there are. This works best, of course, when we don't have to haul around an entire infinite series, but can instead represent it by some more compact function whose expansion gives the desired series. Such a function is called a **generating function**, and manipulating generating functions can be a powerful alternative to creativity in making combinatorial arguments.

11.3.1.3 Formal definition

Given a sequence $a_0, a_1, a_2, \ldots$, its **generating function** $F(z)$ is given by the sum

$$F(z) = \sum_{i=0}^{\infty} a_i z^i.$$

A sum in this form is called a **formal power series**. It is "formal" in the sense that we don't necessarily plan to actually compute the sum, and are instead using the string of z^i terms as a long rack to store coefficients on.

In some cases, the sum has a more compact representation. For example, we have

$$\frac{1}{1-z} = \sum_{i=0}^{\infty} z^i,$$

so $1/(1-z)$ is the generating function for the sequence $1, 1, 1, \ldots$. This may let us manipulate this sequence conveniently by manipulating the generating function.

Here's a simple case. If $F(z)$ generates some sequence a_i, what does sequence b_i does $F(2z)$ generate? The i-th term in the expansion of $F(2z)$ will be $a_i(2z)^i = a_i 2^i z^i$, so we have $b_i = 2^i a_i$. This means that the sequence $1, 2, 4, 8, 16, \ldots$ has generating function $1/(1-2z)$. In general, if $F(z)$ represents a_i, then $F(cz)$ represents $c^i a_i$.

What else can we do to F? One useful operation is to take its derivative with respect to z. We then have

$$\frac{d}{dz}F(z) = \sum_{i=0}^{\infty} a_i \frac{d}{dz} z^i = \sum_{i=0}^{\infty} a_i i z^{i-1}.$$

This *almost* gets us the representation for the series ia_i, but the exponents on the z's are off by one. But that's easily fixed:

$$z\frac{d}{dz}F(z) = z\sum_{i=0}^{\infty} a_i i z^{i-1} = \sum_{i=0}^{\infty} a_i i z^i.$$

So the sequence $0, 1, 2, 3, 4, \ldots$ has generating function

$$z\frac{d}{dz}\frac{1}{1-z} = \frac{z}{(1-z)^2},$$

and the sequence of squares $0, 1, 4, 9, 16, \ldots$ has generating function

$$z\frac{d}{dz}\frac{z}{(1-z)^2} = \frac{z}{(1-z)^2} + \frac{2z^2}{(1-z)^3}.$$

As you can see, some generating functions are prettier than others.

(We can also use integration to divide each term by i, but the details are messier.)

Another way to get the sequence $0, 1, 2, 3, 4, \ldots$ is to observe that it satisfies the recurrence:

- $a_0 = 0$.

- $a_{n+1} = a_n + 1 (\forall n \in \mathbb{N})$.

A standard trick in this case is to multiply each of the $\forall i$ bits by z^n, sum over all n, and see what happens. This gives $\sum a_{n+1} z^n = \sum a_n z^n + \sum z^n = \sum a_n z^n + 1/(1-z)$. The first term on the right-hand side is the generating function for a_n, which we can call $F(z)$ so we don't have to keep writing it out. The second term is just the generating function for $1, 1, 1, 1, 1, \ldots$. But what about the left-hand side? This is almost the same as $F(z)$, except the coefficients don't match up with the exponents. We can fix this by dividing $F(z)$ by z, after carefully subtracting off the a_0 term:

$$\begin{aligned}
(F(z) - a_0)/z &= \left(\sum_{n=0}^{\infty} a_n z^n - a_0 \right) /z \\
&= \left(\sum_{n=1}^{\infty} a_n z^n \right) /z \\
&= \sum_{n=1}^{\infty} a_n z^{n-1} \\
&= \sum_{n=0}^{\infty} a_{n+1} z^n.
\end{aligned}$$

So this gives the equation $(F(z) - a_0)/z = F(z) + 1/(1-z)$. Since $a_0 = 0$, we can rewrite this as $F(z)/z = F(z) + 1/(1-z)$. A little bit of algebra turns this into $F(z) - zF(z) = z/(1-z)$ or $F(z) = z/(1-z)^2$.

Yet another way to get this sequence is construct a collection of objects with a simple structure such that there are exactly n objects with weight n. One way to do this is to consider strings of the form $a^+ b^*$ where we have at least one a followed by zero or more b's. This gives n strings of length n, because we get one string for each of the 1 through n a's we can put in (an example would be abb, aab, and aaa for $n = 3$). We can compute the generating function for this set because to generate each string we must pick in order:

- One initial a. Generating function $= z$.
- Zero or more a's. Generating function $= 1/(1-z)$.
- Zero or more b's. Generating function $= 1/(1-z)$.

Taking the product of these gives $z/(1-z)^2$, as before.

This trick is useful in general; if you are given a generating function $F(z)$ for a_n, but want a generating function for $b_n = \sum_{k \le n} a_k$, allow yourself to

pad each weight-k object out to weight n in exactly one way using $n-k$ junk objects, i.e. multiply $F(z)$ by $1/(1-z)$.

11.3.2 Some standard generating functions

Here is a table of some of the most useful generating functions.

$$\begin{aligned}
\frac{1}{1-z} &= \sum_{i=0}^{\infty} z^i \\
\frac{z}{(1-z)^2} &= \sum_{i=0}^{\infty} iz^i \\
(1+z)^n &= \sum_{i=0}^{\infty} \binom{n}{i} z^i = \sum_{i=0}^{n} \binom{n}{i} z^i \\
\frac{1}{(1-z)^n} &= \sum_{i=0}^{\infty} \binom{n+i-1}{i} z^i
\end{aligned}$$

Of these, the first is the most useful to remember (it's also handy for remembering how to sum geometric series). All of these equations can be proven using the binomial theorem.

11.3.3 More operations on formal power series and generating functions

Let $F(z) = \sum_i a_i z^i$ and $G(z) = \sum_i b_i z^i$. Then their sum $F(z) + G(z) = \sum_i (a_i + b_i) z^i$ is the generating function for the sequence $(a_i + b_i)$. What is their product $F(z)G(z)$?

To compute the i-th term of $F(z)G(z)$, we have to sum over all pairs of terms, one from F and one from G, that produce a z^i factor. Such pairs of terms are precisely those that have exponents that sum to i. So we have

$$F(z)G(z) = \sum_{i=0}^{\infty} \left(\sum_{j=0}^{i} a_j b_{j-i} \right) z^i.$$

As we've seen, this equation has a natural combinatorial interpretation. If we interpret the coefficient a_i on the i-th term of $F(z)$ as counting the number of "a-things" of weight i, and the coefficient b_i as the number of "b-things" of weight i, then the i-th coefficient of $F(z)G(z)$ counts the number of ways to make a combined thing of total weight i by gluing together an a-thing and a b-thing.

As a special case, if $F(z) = G(z)$, then the i-th coefficient of $F(z)G(z) = F^2(z)$ counts how many ways to make a thing of total weight i using two "a-things", and $F^n(z)$ counts how many ways (for each i) to make a thing of total weight i using n "a-things". This gives us an easy combinatorial proof of a special case of the binomial theorem:

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

Think of the left-hand side as the generating function $F(x) = 1 + x$ raised to the n-th power. The function F by itself says that you have a choice between one weight-0 object or one weight-1 object. On the right-hand side the i-th coefficient counts how many ways you can put together a total of i weight-1 objects given n to choose from—so it's $\binom{n}{i}$.

11.3.4 Counting with generating functions

The product formula above suggests that generating functions can be used to count combinatorial objects that are built up out of other objects, where our goal is to count the number of objects of each possible non-negative integer "weight" (we put "weight" in scare quotes because we can make the "weight" be any property of the object we like, as long as it's a non-negative integer—a typical choice might be the size of a set, as in the binomial theorem example above). There are five basic operations involved in this process; we've seen two of them already, but will restate them here with the others.

Throughout this section, we assume that $F(z)$ is the generating function counting objects in some set A and $G(z)$ the generating function counting objects in some set B.

11.3.4.1 Disjoint union

Suppose $C = A \cup B$ and A and B are disjoint. Then the generating function for objects in C is $F(z) + G(z)$.

Example: Suppose that A is the set of all strings of zero or more letters x, where the weight of a string is just its length. Then $F(z) = 1/(1-z)$, since there is exactly one string of each length and the coefficient a_i on each z^i is always 1. Suppose that B is the set of all strings of zero or more letters y and/or z, so that $G(z) = 1/(1-2z)$ (since there are now 2^i choices of length-i strings). The set C of strings that are either (a) all x's or (b) made up of y's, z's, or both, has generating function $F(z) + G(z) = 1/(1-z) + 1/(1-2z)$.

11.3.4.2 Cartesian product

Now let $C = A \times B$, and let the weight of a pair $(a, b) \in C$ be the sum of the weights of a and b. Then the generating function for objects in C is $F(z)G(z)$.

Example: Let A be all-x strings and B be all-y or all-z strings, as in the previous example. Let C be the set of all strings that consist of zero or more x's followed by zero or more y's and/or z's. Then the generating function for C is $F(z)G(z) = \frac{1}{(1-z)(1-2z)}$.

11.3.4.3 Repetition

Now let C consists of all finite sequences of objects in A, with the weight of each sequence equal to the sum of the weights of its elements (0 for an empty sequence). Let $H(z)$ be the generating function for C. From the preceding rules we have

$$H = 1 + F + F^2 + F^3 + \cdots = \frac{1}{1 - F}.$$

This works best when $H(0) = 0$; otherwise we get infinitely many weight-0 sequences. It's also worth noting that this is just a special case of substitution (see below), where our "outer" generating function is $1/(1 - z)$.

Example: $(0|11)*$ Let $A = \{0, 11\}$, and let C be the set of all sequences of zeros and ones where ones occur only in even-length runs. Then the generating function for A is $z + z^2$ and the generating function for C is $1/(1-z-z^2)$. We can extract exact coefficients from this generating function using the techniques below.

Example: sequences of positive integers Suppose we want to know how many different ways there are to generate a particular integer as a sum of positive integers. For example, we can express 4 as 4, $3+1$, $2+2$, $2+1+1$, $1+1+1+1$, $1+1+2$, $1+2+1$, or $1+3$, giving 8 different ways.

We can solve this problem using the repetition rule. Let $F = z/(1 - z)$

generate all the positive integers. Then

$$\begin{aligned} H &= \frac{1}{1-F} \\ &= \frac{1}{1-\frac{z}{1-z}} \\ &= \frac{1-z}{(1-z)-z} \\ &= \frac{1-z}{1-2z}. \end{aligned}$$

We can get exact coefficients by observing that

$$\begin{aligned} \frac{1-z}{1-2z} &= \frac{1}{1-2z} - \frac{z}{1-2z} \\ &= \sum_{n=0}^{\infty} 2^n z^n - \sum_{n=0}^{\infty} 2^n z^{n+1} \\ &= \sum_{n=0}^{\infty} 2^n z^n - \sum_{n=1}^{\infty} 2^{n-1} z^n \\ &= 1 + \sum_{n=1}^{\infty} (2^n - 2^{n-1}) z^n \\ &= 1 + \sum_{n=1}^{\infty} 2^{n-1} z^n. \end{aligned}$$

This means that there is 1 way to express 0 (the empty sum), and 2^{n-1} ways to express any larger value n (e.g. $2^{4-1} = 8$ ways to express 4).

Once we know what the right answer is, it's not terribly hard to come up with a combinatorial explanation. The quantity 2^{n-1} counts the number of subsets of an $(n-1)$-element set. So imagine that we have $n-1$ places and we mark some subset of them, plus add an extra mark at the end; this might give us a pattern like `XX-X`. Now for each sequence of places ending with a mark we replace it with the number of places (e.g. `XX-X` $= 1, 1, 2$, `X--X-X---X` $= 1, 3, 2, 4$). Then the sum of the numbers we get is equal to n, because it's just counting the total length of the sequence by dividing it up at the marks and the adding the pieces back together. The value 0 doesn't fit this pattern (we can't put in the extra mark without getting a sequence of length 1), so we have 0 as a special case again.

If we are very clever, we might come up with this combinatorial explanation from the beginning. But the generating function approach saves us from having to be clever.

11.3.4.4 Pointing

This operation is a little tricky to describe. Suppose that we can think of each weight-k object in A as consisting of k items, and that we want to count not only how many weight-k objects there are, but how many ways we can produce a weight-k object where one of its k items has a special mark on it. Since there are k different items to choose for each weight-k object, we are effectively multiplying the count of weight-k objects by k. In generating function terms, we have

$$H(z) = z\frac{d}{dz}F(z).$$

Repeating this operation allows us to mark more items (with some items possibly getting more than one mark). If we want to mark n distinct items in each object (with distinguishable marks), we can compute

$$H(z) = z^n\frac{d^n}{dz^n}F(z),$$

where the repeated derivative turns each term $a_i z^i$ into $a_i i(i-1)(i-2)\dots(i-n+1)z^{i-n}$ and the z^n factor fixes up the exponents. To make the marks indistinguishable (i.e., we don't care what order the values are marked in), divide by $n!$ to turn the extra factor into $\binom{i}{n}$.

(If you are not sure how to take a derivative, look at §H.2.)

Example: Count the number of finite sequences of zeros and ones where exactly two digits are underlined. The generating function for $\{0,1\}$ is $2z$, so the generating function for sequences of zeros and ones is $F = 1/(1-2z)$ by the repetition rule. To mark two digits with indistinguishable marks, we need to compute

$$\frac{1}{2}z^2\frac{d^2}{dz^2}\frac{1}{1-2z} = \frac{1}{2}z^2\frac{d}{dz}\frac{2}{(1-2z)^2} = \frac{1}{2}z^2\frac{8}{(1-2z)^3} = \frac{4z^2}{(1-2z)^3}.$$

11.3.4.5 Substitution

Suppose that the way to make a C-thing is to take a weight-k A-thing and attach to each its k items a B-thing, where the weight of the new C-thing is the sum of the weights of the B-things. Then the generating function for C is the composition $F(G(z))$.

Why this works: Suppose we just want to compute the number of C-things of each weight that are made from some single specific weight-k A-thing. Then the generating function for this quantity is just $(G(z))^k$. If we expand our horizons to include all a_k weight-k A-things, we have to multiply by a_k

to get $a_k(G(z))^k$. If we further expand our horizons to include A-things of all different weights, we have to sum over all k:

$$\sum_{k=0}^{\infty} a_k(G(z))^k.$$

But this is just what we get if we start with $F(z)$ and substitute $G(z)$ for each occurrence of z, i.e. if we compute $F(G(z))$.

Example: bit-strings with primes Suppose we let A be all sequences of zeros and ones, with generating function $F(z) = 1/(1-2z)$. Now suppose we can attach a single or double prime to each 0 or 1, giving $0'$ or $0''$ or $1'$ or $1''$, and we want a generating function for the number of distinct primed bit-strings with n attached primes. The set $\{', ''\}$ has generating function $G(z) = z + z^2$, so the composite set has generating function $F(z) = 1/(1-2(z+z^2)) = 1/(1-2z-2z^2)$.

Example: (0|11)* again The previous example is a bit contrived. Here's one that's a little more practical, although it involves a brief digression into **multivariate generating functions**. A multivariate generating function $F(x,y)$ generates a series $\sum_{ij} a_{ij}x^i y^j$, where a_{ij} counts the number of things that have i x's and j y's. (There is also the obvious generalization to more than two variables). Consider the multivariate generating function for the set $\{0,1\}$, where x counts zeros and y counts ones: this is just $x+y$. The multivariate generating function for sequences of zeros and ones is $1/(1-x-y)$ by the repetition rule. Now suppose that each 0 is left intact but each 1 is replaced by 11, and we want to count the total number of strings by length, using z as our series variable. So we substitute z for x and z^2 for y (since each y turns into a string of length 2), giving $1/(1-z-z^2)$. This gives another way to get the generating function for strings built by repeating 0 and 11.

11.3.5 Generating functions and recurrences

What makes generating functions particularly useful for algorithm analysis is that they directly solve recurrences of the form $T(n) = aT(n-1) + bT(n-2) + f(n)$ (or similar recurrences with more T terms on the right-hand side), provided we have a generating function $F(z)$ for $f(n)$. The idea is that there exists some generating function $G(z)$ that describes the entire sequence of values $T(0), T(1), T(2), \ldots$, and we just need to solve for it

by restating the recurrence as an equation about G. The left-hand side will just turn into G. For the right-hand side, we need to shift $T(n-1)$ and $T(n-2)$ to line up right, so that the right-hand side will correctly represent the sequence $T(0), T(1), aT(0) + aT(1) + F(2)$, etc. It's not hard to see that the generating function for the sequence $0, T(0), T(1), T(2), \ldots$ (corresponding to the $T(n-1)$ term) is just $zG(z)$, and similarly the sequence $0, 0, T(1), T(2), T(3), \ldots$ (corresponding to the $T(n-2)$ term) is $z^2G(z)$. So we have (being very careful to subtract out extraneous terms at for $i = 0$ and $i = 1$):

$$G = az(G - T(0)) + bz^2G + (F - f(0) - zf(1)) + T(0) + zT(1),$$

and after expanding F we can in principle solve this for G as a function of z.

11.3.5.1 Example: A Fibonacci-like recurrence

Let's take a concrete example. The Fibonacci-like recurrence

$$T(n) = T(n-1) + T(n-2), T(0) = 1, T(1) = 1,$$

becomes

$$G = (zG - z) + z^2G + 1 + z.$$

(here $F = 0$).
Solving for G gives

$$G = 1/(1 - z - z^2).$$

Unfortunately this is not something we recognize from our table, although it has shown up in a couple of examples. (Exercise: *Why* does the recurrence $T(n) = T(n-1) + T(n-2)$ count the number of strings built from 0 and 11 of length n?) In the next section we show how to recover a closed-form expression for the coefficients of the resulting series.

11.3.6 Recovering coefficients from generating functions

There are basically three ways to recover coefficients from generating functions:

1. Recognize the generating function from a table of known generating functions, or as a simple combination of such known generating functions. This doesn't work very often but it is possible to get lucky.

2. To find the k-th coefficient of $F(z)$, compute the k-th derivative $d^k/dz^k F(z)$ and divide by $k!$ to shift a_k to the z^0 term. Then substitute 0 for z. For example, if $F(z) = 1/(1-z)$ then $a_0 = 1$ (no differentiating), $a_1 = 1/(1-0)^2 = 1$, $a_2 = 1/(1-0)^3 = 1$, etc. This usually only works if the derivatives have a particularly nice form or if you only care about the first couple of coefficients (it's particularly effective if you only want a_0).

3. If the generating function is of the form $1/Q(z)$, where Q is a polynomial with $Q(0) \neq 0$, then it is generally possible to expand the generating function out as a sum of terms of the form $P_c/(1-z/c)$ where c is a root of Q (i.e. a value such that $Q(c) = 0$). Each denominator P_c will be a constant if c is not a repeated root; if c is a repeated root, then P_c can be a polynomial of degree up to one less than the multiplicity of c. We like these expanded solutions because we recognize $1/(1-z/c) = \sum_i c^{-i} z^i$, and so we can read off the coefficients a_i generated by $1/Q(z)$ as an appropriately weighted some of c_1^{-i}, c_2^{-i}, etc., where the c_j range over the roots of Q.

Example: Take the generating function $G = 1/(1-z-z^2)$. We can simplify it by factoring the denominator: $1-z-z^2 = (1-az)(1-bz)$ where $1/a$ and $1/b$ are the solutions to the equation $1-z-z^2 = 0$; in this case $a = (1+\sqrt{5})/2$, which is approximately 1.618 and $b = (1-\sqrt{5})/2$, which is approximately -0.618. It happens to be the case that we can always expand $1/P(z)$ as $A/(1-az) + B/(1-bz)$ for some constants A and B whenever P is a degree 2 polynomial with constant coefficient 1 and distinct roots a and b, so

$$G = \frac{A}{1-az} + \frac{B}{1-bz},$$

and here we can recognize the right-hand side as the sum of the generating functions for the sequences $A \cdot a^i$ and $B \cdot b^i$. The $A \cdot a^i$ term dominates, so we have that $T(n) = \Theta(a^n)$, where a is approximately 1.618. We can also solve for A and B exactly to find an exact solution if desired.

A rule of thumb that applies to recurrences of the form $T(n) = a_1 T(n-1) + a_2 T(n-2) + \ldots a_k T(n-k) + f(n)$ is that unless f is particularly large, the solution is usually exponential in $1/x$, where x is the smallest root of the polynomial $1 - a_1 z - a_2 z^2 \cdots - a_k z^k$. This can be used to get very quick estimates of the solutions to such recurrences (which can then be proved without fooling around with generating functions).

Exercise: What is the exact solution if $T(n) = T(n-1) + T(n-2) + 1$? Or if $T(n) = T(n-1) + T(n-2) + n$?

11.3.6.1 Partial fraction expansion and Heaviside's cover-up method

There is a nice trick for finding the numerators in a partial fraction expansion. Suppose we have

$$\frac{1}{(1-az)(1-bz)} = \frac{A}{1-az} + \frac{B}{1-bz}.$$

Multiply both sides by $1-az$ to get

$$\frac{1}{1-bz} = A + \frac{B(1-az)}{1-bz}.$$

Now plug in $z = 1/a$ to get

$$\frac{1}{1-b/a} = A + 0.$$

We can immediately read off A. Similarly, multiplying by $1-bz$ and then setting $1-bz$ to zero gets B. The method is known as the "cover-up method" because multiplication by $1-az$ can be simulated by covering up $1-az$ in the denominator of the left-hand side and all the terms that don't have $1-az$ in the denominator in the right hand side.

The cover-up method will work in general whenever there are no repeated roots, even if there are many of them; the idea is that setting $1-qz$ to zero knocks out all the terms on the right-hand side but one. With repeated roots we have to worry about getting numerators that aren't just a constant, so things get more complicated. We'll come back to this case below.

Example: A simple recurrence Suppose $f(0) = 0, f(1) = 1$, and for $n \geq 2$, $f(n) = f(n-1) + 2f(n-2)$. Multiplying these equations by z^n and summing over all n gives a generating function

$$F(z) = \sum_{n=0} \infty f(n) z^n = 0 \cdot z^0 + 1 \cdot z^1 + \sum_{n=2}^{\infty} f(n-1) z^n + \sum_{n=2}^{\infty} 2f(n-2) z^n.$$

With a bit of tweaking, we can get rid of the sums on the RHS by

converting them into copies of F:

$$\begin{aligned}
F(z) &= z + \sum_{n=2}^{\infty} f(n-1)z^n + 2\sum_{n=2}^{\infty} f(n-2)z^n \\
&= z + \sum_{n=1}^{\infty} f(n)z^{n+1} + 2\sum_{n=0}^{\infty} f(n)z^{n+2} \\
&= z + z\sum_{n=1}^{\infty} f(n)z^n + 2z^2\sum_{n=0}^{\infty} f(n)z^n \\
&= z + z(F(z) - f(0)z^0) + 2z^2F(z) \\
&= z + zF(z) + 2z^2F(z).
\end{aligned}$$

Now solve for $F(z)$ to get $F(x) = \frac{z}{1-z-2z^2} = \frac{z}{(1+z)(1-2z)} = z\left(\frac{A}{1+z} + \frac{B}{1-2z}\right)$, where we need to solve for A and B.

We can do this directly, or we can use the cover-up method. The cover-up method is easier. Setting $z = -1$ and covering up $1 + z$ gives $A = 1/(1 - 2(-1)) = 1/3$. Setting $z = 1/2$ and covering up $1 - 2z$ gives $B = 1/(1 + z) = 1/(1 + 1/2) = 2/3$. So we have

$$\begin{aligned}
F(z) &= \frac{(1/3)z}{1+z} + \frac{(2/3)z}{1-2z} \\
&= \sum_{n=0}^{\infty} \frac{(-1)^n}{3} z^{n+1} + \sum_{n=0}^{\infty} \frac{2\cdot 2^n}{3} z^{n+1} \\
&= \sum_{n=1}^{\infty} \frac{(-1)^{n-1}}{3} z^n + \sum_{n=1}^{\infty} \frac{2^n}{3} z^n \\
&= \sum_{n=1}^{\infty} \left(\frac{2^n - (-1)^n}{3}\right) z^n.
\end{aligned}$$

This gives $f(0) = 0$ and, for $n \geq 1$, $f(n) = \frac{2^{n-(-1)n}}{3}$. It's not hard to check that this gives the same answer as the recurrence.

Example: Coughing cows Let's count the number of strings of each length of the form `(M)*(O|U)*(G|H|K)*` where `(x|y)` means we can use x or y and `*` means we can repeat the previous parenthesized expression 0 or more times (these are examples of **regular expressions**).

We start with a sequence of 0 or more M's. The generating function for this part is our old friend $1/(1-z)$. For the second part, we have two choices for each letter, giving $1/(1-2z)$. For the third part, we have $1/(1-3z)$.

Since each part can be chosen independently of the other two, the generating function for all three parts together is just the product:

$$\frac{1}{(1-z)(1-2z)(1-3z)}.$$

Let's use the cover-up method to convert this to a sum of partial fractions. We have

$$\begin{aligned}\frac{1}{(1-z)(1-2z)(1-3z)} &= \frac{\left(\frac{1}{(1-2)(1-3)}\right)}{1-z} + \frac{\left(\frac{1}{(1-\frac{1}{2})(1-\frac{3}{2})}\right)}{1-2z} + \frac{\left(\frac{1}{(1-\frac{1}{3})(1-\frac{2}{3})}\right)}{1-3z} \\ &= \frac{\frac{1}{2}}{1-z} + \frac{-4}{1-2z} + \frac{\frac{9}{2}}{1-3z}.\end{aligned}$$

So the exact number of length-n sequences is $(1/2) - 4 \cdot 2^n + (9/2) \cdot 3^n$. We can check this for small n:

n	Formula	Strings
0	$1/2 - 4 + 9/2 = 1$	()
1	$1/2 - 8 + 27/2 = 6$	M, O, U, G, H, K
2	$1/2 - 16 + 81/2 = 25$	$MM, MO, MU, MG, MH, MK, OO, OU, OG, OH, OK, UO,$ $UU, UG, UH, UK, GG, GH, GK, HG, HH, HK, KG, KH, KK$
3	$1/2 - 32 + 243/2 = 90$	(exercise)$\ddot\smile$

Example: A messy recurrence Let's try to solve the recurrence $T(n) = 4T(n-1) + 12T(n-2) + 1$ with $T(0) = 0$ and $T(1) = 1$.

Let $F = \sum T(n)z^n$.

Summing over all n gives

$$\begin{aligned}F = \sum_{n=0}^{\infty} T(n)z^n &= T(0)z^0 + T(1)z^1 + 4\sum_{n=2}^{\infty} T(n-1)z^n + 12\sum_{n=2}^{\infty} T(n-2)z^n + \sum_{n=2}^{\infty} 1 \cdot z^n \\ &= z + 4z\sum_{n=1}^{\infty} T(n)z^n + 12z^2\sum_{n=0}^{\infty} T(n)z^n + z^2\sum_{n=0}^{\infty} z^n \\ &= z + 4z(F - T(0)) + 12z^2F + \frac{z^2}{1-z} \\ &= z + 4zF + 12z^2F + \frac{z^2}{1-z}.\end{aligned}$$

Solving for F gives

$$F = \frac{\left(z + \frac{z^2}{1-z}\right)}{1 - 4z - 12z^2}.$$

We want to solve this using partial fractions, so we need to factor $(1 - 4z - 12z^2) = (1 + 2z)(1 - 6z)$. This gives

$$\begin{aligned}
F &= \frac{\left(z + \frac{z^2}{1-z}\right)}{(1+2z)(1-6z)} \\
&= \frac{z}{(1+2z)(1-6z)} + \frac{z^2}{(1-z)(1+2z)(1-6z)}. \\
&= z\left(\frac{1}{(1+2z)\left(1-6\left(-\frac{1}{2}\right)\right)} + \frac{1}{\left(1+2\left(\frac{1}{6}\right)\right)(1-6z)}\right) \\
&+ z^2\left(\frac{1}{(1-z)(1+2)(1-6)} + \frac{1}{\left(1-\left(-\frac{1}{2}\right)\right)(1+2z)\left(1-6\left(-\frac{1}{2}\right)\right)} + \frac{1}{\left(1-\frac{1}{6}\right)\left(1+2\left(\frac{1}{6}\right)\right)(1-6z)}\right) \\
&= \frac{\frac{1}{4}z}{1+2z} + \frac{\frac{3}{4}z}{1-6z} + \frac{-\frac{1}{15}z^2}{1-z} + \frac{\frac{1}{6}z^2}{1+2z} + \frac{\frac{9}{10}z^2}{1-6z}.
\end{aligned}$$

From this we can immediately read off the value of $T(n)$ for $n \geq 2$:

$$\begin{aligned}
T(n) &= \frac{1}{4}(-2)^{n-1} + \frac{3}{4}6^{n-1} - \frac{1}{15} + \frac{1}{6}(-2)^{n-2} + \frac{9}{10}6^{n-2} \\
&= -\frac{1}{8}(-2)^n + \frac{1}{8}6^n - \frac{1}{15} + \frac{1}{24}(-2)^n + \frac{1}{40}6^n \\
&= \frac{3}{20}6^n - \frac{1}{12}(-2)^n - \frac{1}{15}.
\end{aligned}$$

Let's check this against the solutions we get from the recurrence itself:

n	$T(n)$
0	0
1	1
2	$1 + 4 \cdot 1 + 12 \cdot 0 = 5$
3	$1 + 4 \cdot 5 + 12 \cdot 1 = 33$
4	$1 + 4 \cdot 33 + 12 \cdot 5 = 193$

We'll try $n = 3$, and get $T(3) = (3/20) \cdot 216 + 8/12 - 1/15 = (3 \cdot 3 \cdot 216 + 40 - 4)/60 = (1944 + 40 - 4)/60 = 1980/60 = 33$.

To be extra safe, let's try $T(2) = (3/20) \cdot 36 - 4/12 - 1/15 = (3 \cdot 3 \cdot 36 - 20 - 4)/60 = (324 - 20 - 4)/60 = 300/60 = 5$. This looks good too.

The moral of this exercise? Generating functions can solve ugly-looking recurrences exactly, but you have to be very very careful in doing the math.

11.3.6.2 Partial fraction expansion with repeated roots

Let $a_n = 2a_{n-1} + n$, with some constant a_0. We'd like to find a closed-form formula for a_n.

As a test, let's figure out the first few terms of the sequence:

$$\begin{aligned}
a_0 &= a_0 \\
a_1 &= 2a_0 + 1 \\
a_2 &= 4a_0 + 2 + 2 = 4a_0 + 4 \\
a_3 &= 8a_0 + 8 + 3 = 8a_0 + 11 \\
a_4 &= 16a_0 + 22 + 4 = 16a_0 + 26
\end{aligned}$$

The a_0 terms look nice (they're $2^n a_0$), but the 0, 1, 4, 11, 26 sequence doesn't look like anything familiar. So we'll find the formula the hard way.

First we convert the recurrence into an equation over generating functions and solve for the generating function F:

$$\begin{aligned}
\sum a_n z^n &= 2\sum a_{n-1} z^n + \sum n z^n + a_0 \\
F &= 2zF + \frac{z}{(1-z)^2} + a_0 \\
(1-2z)F &= \frac{z}{(1-z)^2} + a_0 \\
F &= \frac{z}{(1-z)^2(1-2z)} + \frac{a_0}{1-2z}.
\end{aligned}$$

Observe that the right-hand term gives us exactly the $2^n a_0$ terms we expected, since $1/(1-2z)$ generates the sequence 2^n. But what about the left-hand term? Here we need to apply a partial-fraction expansion, which is simplified because we already know how to factor the denominator but is complicated because there is a repeated root.

We can now proceed in one of two ways: we can solve directly for the partial fraction expansion, or we can use an extended version of Heaviside's cover-up method that handles repeated roots using differentiation. We'll start with the direct method.

Solving for the PFE directly Write

$$\frac{1}{(1-z)^2(1-2z)} = \frac{A}{(1-z)^2} + \frac{B}{1-2z}$$

.

We expect B to be a constant and A to be of the form $A_1 z + A_0$.

To find B, use the technique of multiplying by $1-2z$ and setting $z = 1/2$:

$$\frac{1}{(1-\frac{1}{2})^2} = \frac{A \cdot 0}{(1-z)^2} + B.$$

So $B = 1/(1-1/2)^2 = 1/(1/4) = 4$.
We can't do this for A, but we can solve for it after substituting in $B = 4$:

$$\begin{aligned}
\frac{1}{(1-z)^2(1-2z)} &= \frac{A}{(1-z)^2} + \frac{4}{1-2z} \\
1 &= A(1-2z) + 4(1-z)^2 \\
A &= \frac{1-4(1-z)^2}{1-2z} \\
&= \frac{1-4+8z-4z^2}{1-2z} \\
&= \frac{-3+8z-4z^2}{1-2z} \\
&= \frac{-(1-2z)(3-2z)}{1-2z} \\
&= 2z-3.
\end{aligned}$$

So we have the expansion

$$\frac{1}{(1-z)^2(1-2z)} = \frac{2z-3}{(1-z)^2} + \frac{4}{1-2z},$$

from which we get

$$\begin{aligned}
F &= \frac{z}{(1-z)^2(1-2z)} + \frac{a_0}{1-2z} \\
&= \frac{2z^2-3z}{(1-z)^2} + \frac{4z}{1-2z} + \frac{a_0}{1-2z}.
\end{aligned}$$

If we remember that $1/(1-z)^2$ generates the sequence $x_n = n+1$ and $1/(1-2z)$ generates $x_n = 2^n$, then we can quickly read off the solution (for large n):

$$a_n = 2(n-1) - 3n + 4 \cdot 2^{n-1} + a_0 \cdot 2^n = 2^n a_0 + 2^{n+1} - 2 - n$$

which we can check by plugging in particular values of n and comparing it to the values we got by iterating the recurrence before.

The reason for the "large n" caveat is that $z^2/(1-z)^2$ doesn't generate precisely the sequence $x_n = n-1$, since it takes on the values $0, 0, 1, 2, 3, 4, \ldots$ instead of $-1, 0, 1, 2, 3, 4, \ldots$. Similarly, the power series for $z/(1-2z)$ does not have the coefficient $2^{n-1} = 1/2$ when $n = 0$. Miraculously, in this particular example the formula works for $n = 0$, even though it shouldn't: $2(n-1)$ is -2 instead of 0, but $4 \cdot 2^{n-1}$ is 2 instead of 0, and the two errors cancel each other out.

Solving for the PFE using the extended cover-up method It is also possible to extend the cover-up method to handle repeated roots. Here we choose a slightly different form of the partial fraction expansion:

$$\frac{1}{(1-z)^2(1-2z)} = \frac{A}{(1-z)^2} + \frac{B}{1-z} + \frac{C}{1-2z}.$$

Here A, B, and C are all constants. We can get A and C by the cover-up method, where for A we multiply both sides by $(1-z)^2$ before setting $z = 1$; this gives $A = 1/(1-2) = -1$ and $C = 1/(1-\frac{1}{2})^2 = 4$. For B, if we multiply both sides by $(1-z)$ we are left with $A/(1-z)$ on the right-hand side and a $(1-z)$ in the denominator on the left-hand side. Clearly setting $z = 1$ in this case will not help us.

The solution is to first multiply by $(1-z)^2$ as before but then take a derivative:

$$\begin{aligned}
\frac{1}{(1-z)^2(1-2z)} &= \frac{A}{(1-z)^2} + \frac{B}{1-z} + \frac{C}{1-2z} \\
\frac{1}{1-2z} &= A + B(1-z) + \frac{C(1-z)^2}{1-2z} \\
\frac{d}{dz}\frac{1}{1-2z} &= \frac{d}{dz}\left(A + B(1-z) + \frac{C(1-z)^2}{1-2z}\right) \\
\frac{2}{(1-2z)^2} &= -B + \frac{-2C(1-z)}{1-2z} + \frac{2C(1-z)^2}{(1-2z)^2}
\end{aligned}$$

Now if we set $z = 1$, every term on the right-hand side except $-B$ becomes 0, and we get $-B = 2/(1-2)^2$ or $B = -2$.

Plugging A, B, and C into our original formula gives

$$\frac{1}{(1-z)^2(1-2z)} = \frac{-1}{(1-z)^2} + \frac{-2}{1-z} + \frac{4}{1-2z},$$

and thus

$$F = \frac{z}{(1-z)^2(1-2z)} + \frac{a_0}{1-2z} = z\left(\frac{-1}{(1-z)^2} + \frac{-2}{1-z} + \frac{4}{1-2z}\right) + \frac{a_0}{1-2z}.$$

From this we can read off (for large n):

$$a_n = 4 \cdot 2^{n-1} - n - 2 + a_0 \cdot 2^n = 2^{n+1} + 2^n a_0 - n - 2.$$

We believe this because it looks like the solution we already got.

11.3.7 Asymptotic estimates

We can simplify our life considerably if we only want an asymptotic estimate of a_n (see Chapter 7). The basic idea is that if a_n is non-negative for sufficiently large n and $\sum a_n z^n$ converges for some fixed value z, then a_n must be $o(z^{-n})$ in the limit. (Proof: otherwise, $a_n z^n$ is at least a constant for infinitely many n, giving a divergent sum.) So we can use the **radius of convergence** of a generating function $F(z)$, defined as the largest value r such that $F(z)$ is defined for all (complex) z with $|z| < r$, to get a quick estimate of the growth rate of F's coefficients: whatever they do, we have $a_n = O(r^{-n})$.

For generating functions that are **rational functions** (ratios of polynomials), we can use the partial fraction expansion to do even better. First observe that for $F(z) = \sum f_i z^n = 1/(1-az)^k$, we have $f_n = \binom{k+n-1}{n} a^n = \frac{(n+k-1)(n+k-2)\ldots(k-1)}{(k-1)!} a^n = \Theta(a^n n^{k-1})$. Second, observe that the numerator is irrelevant: if $1/(1-az)^k = \Theta(a^n n^{k-1})$ then $bz^m/(1-az)^{k-1} = b\Theta(a^{n-m}(n-m)^{k-1}) = ba^{-m}(1-m/n)^{k-1}\Theta(a^n n^{k-1}) = \Theta(a^n n^{k-1})$, because everything outside the Θ disappears into the constant for sufficiently large n. Finally, observe that in a partial fraction expansion, the term $1/(1-az)^k$ with the largest coefficient a (if there is one) wins in the resulting asymptotic sum: $\Theta(a^n) + \Theta(b^n) = \Theta(a^n)$ if $|a| > |b|$. So we have:

Theorem 11.3.1. *Let $F(z) = \sum f_n z^n = P(z)/Q(z)$ where P and Q are polynomials in z. If Q has a root r with multiplicity k, and all other roots s of Q satisfy $|r| < |s|$, then $f_n = \Theta((1/r)^n n^{k-1})$.*

The requirement that r is a unique minimal root of Q is necessary; for example, $F(z) = 2/(1-z^2) = 1/(1-z) + 1/(1+z)$ generates the sequence $0, 2, 0, 2, \ldots$, which is *not* $\Theta(1)$ because of all the zeros; here the problem is that $1 - z^2$ has two roots with the same absolute value, so for some values of n it is possible for them to cancel each other out.

A root in the denominator of a rational function F is called a **pole**. So another way to state the theorem is that the asymptotic value of the coefficients of a rational generating function is determined by the smallest pole.

More examples:

$F(z)$	Smallest pole	Asymptotic value
$1/(1-z)$	1	$\Theta(1)$
$1/(1-z)^2$	1, multiplicity 2	$\Theta(n)$
$1/(1-z-z^2)$	$(\sqrt{5}-1)/2 = 2/(1+\sqrt{5})$	$\Theta(((1+\sqrt{5})/2)^n)$
$1/((1-z)(1-2z)(1-3z))$	1/3	$\Theta(3^n)$
$(z+z^2(1-z))/(1-4z-12z^2)$	1/6	$\Theta(6^n)$
$1/((1-z)^2(1-2z))$	1/2	$\Theta(2^n)$

In each case it may be instructive to compare the asymptotic values to the exact values we obtained earlier.

11.3.8 Recovering the sum of all coefficients

Given a generating function for a convergent series $\sum_i a_i z^i$, we can compute the sum of all the a_i by setting z to 1. Unfortunately, for many common generating functions setting $z = 1$ yields $0/0$ (if it yields something else divided by zero then the series diverges). In this case we can recover the correct sum by taking the limit as z goes to 1 using *L'Hôpital's rule*, which says that $\lim_{x\to c} f(x)/g(x) = \lim_{x\to c} f'(x)/g'(x)$ when the latter limit exists and either $f(c) = g(c) = 0$ or $f(c) = g(c) = \infty$.[10]

11.3.8.1 Example

Let's derive the formula for $1 + 2 + \cdots + n$. We'll start with the generating function for the series $\sum_{i=0}^{n} z^i$, which is $(1 - z^n + 1)/(1 - z)$. Applying the $z\frac{d}{dz}$ method gives us

$$\begin{aligned}\sum_{i=0}^{n} iz^i &= z\frac{d}{dz}\frac{1-z^{n+1}}{1-z} \\ &= z\left(\frac{1}{(1-z)^2} - \frac{(n+1)z^n}{1-z} - \frac{z^{n+1}}{(1-z)^2}\right) \\ &= \frac{z-(n+1)z^{n+1}+nz^{n+2}}{(1-z)^2}.\end{aligned}$$

[10] The justification for doing this is that we know that a finite sequence really has a finite sum, so the "singularity" appearing at $z = 1$ in e.g. $\frac{1-z^{n+1}}{1-z}$ is an artifact of the generating-function representation rather than the original series—it's a "removable singularity" that can be replaced by the limit of $f(x)/g(x)$ as $x \to c$.

Plugging $z = 1$ into this expression gives $(1 - (n+1) + n)/(1-1) = 0/0$, which does not make us happy. So we go to the hospital—twice, since one application of L'Hôpital's rule doesn't get rid of our $0/0$ problem:

$$\begin{aligned}
\lim_{z\to 1} \frac{z - (n+1)z^{n+1} + nz^{n+2}}{(1-z)^2} &= \lim_{z\to 1} \frac{1 - (n+1)^2 z^n + n(n+2)z^{n+1}}{-2(1-z)} \\
&= \lim_{z\to 1} \frac{-n(n+1)^2 z^{n-1} + n(n+1)(n+2)z^n}{2} \\
&= \frac{-n(n+1)^2 + n(n+1)(n+2)}{2} \\
&= \frac{-n^3 - 2n^2 - n + n^3 + 3n^2 + 2n}{2} \\
&= \frac{n^2 + n}{2} = \frac{n(n+1)}{2},
\end{aligned}$$

which is our usual formula. Gauss's childhood proof is a lot quicker, but the generating-function proof is something that we could in principle automate most of the work using a computer algebra system, and it doesn't require much creativity or intelligence. So it might be the weapon of choice for nastier problems where no clever proof comes to mind.

More examples of this technique can be found in §11.2, where the binomial theorem applied to $(1+x)^n$ (which is really just a generating function for $\sum \binom{n}{i} z^i$) is used to add up various sums of binomial coefficients.

11.3.9 A recursive generating function

Let's suppose we want to count binary trees with n internal nodes. We can obtain such a tree either by (a) choosing an empty tree (g.f.: $z^0 = 1$); or (b) choosing a root with weight 1 (g.f. $1 \cdot z^1 = z$), since we can choose it in exactly one way), and two subtrees (g.f. $= F^2$ where F is the g.f. for trees). This gives us a recursive definition

$$F = 1 + zF^2.$$

Solving for F using the quadratic formula gives

$$F = \frac{1 \pm \sqrt{1 - 4z}}{2z}.$$

That $2z$ in the denominator may cause us trouble later, but let's worry about that when the time comes. First we need to figure out how to extract coefficients from the square root term.

The binomial theorem says

$$\sqrt{1-4z} = (1-4z)^{1/2} = \sum_{n=0}^{\infty} \binom{1/2}{n} (-4z)^n.$$

For $n \geq 1$, we can expand out the $\binom{1/2}{n}$ terms as

$$\begin{aligned}
\binom{1/2}{n} &= \frac{(1/2)_n}{n!} \\
&= \frac{1}{n!} \cdot \prod_{k=0}^{n-1} (1/2 - k) \\
&= \frac{1}{n!} \cdot \prod_{k=0}^{n-1} \frac{1-2k}{2} \\
&= \frac{(-1)^n}{2^n n!} \cdot \prod_{k=0}^{n-1} (2k-1) \\
&= \frac{(-1)^n}{2^n n!} \cdot \frac{\prod_{k=1}^{2n-2} k}{\prod_{k=1}^{n-1} 2k} \\
&= \frac{(-1)^n}{2^n n!} \cdot \frac{(2n-2)!}{2^{n-1}(n-1)!} \\
&= \frac{(-1)^n}{2^{2n-1}} \cdot \frac{(2n-2)!}{n!(n-1)!} \\
&= \frac{(-1)^n}{2^{2n-1}(2n-1)} \cdot \frac{(2n-1)!}{n!(n-1)!} \\
&= \frac{(-1)^n}{2^{2n-1}(2n-1)} \cdot \binom{2n-1}{n}.
\end{aligned}$$

For $n = 0$, the switch from the big product of odd terms to $(2n-2)!$ divided by the even terms doesn't work, because $(2n-2)!$ is undefined. So here we just use the special case $\binom{1/2}{0} = 1$.

Now plug this nasty expression back into F to get

$$\begin{aligned}
F &= \frac{1 \pm \sqrt{1-4z}}{2z} \\
&= \frac{1}{2z} \pm \frac{1}{2z} \sum_{n=0}^{\infty} \binom{1/2}{n} (-4z)^n \\
&= \frac{1}{2z} \pm \left(\frac{1}{2z} + \frac{1}{2z} \sum_{n=1}^{\infty} \frac{(-1)^{n-1}}{2^{2n-1}(2n-1)} \binom{2n-1}{n} (-4z)^n \right) \\
&= \frac{1}{2z} \pm \left(\frac{1}{2z} + \frac{1}{2z} \sum_{n=1}^{\infty} \frac{(-1)^{2n-1} 2^{2n}}{2^{2n-1}(2n-1)} \binom{2n-1}{n} z^n \right) \\
&= \frac{1}{2z} \pm \left(\frac{1}{2z} + \frac{1}{2z} \sum_{n=1}^{\infty} \frac{-2}{(2n-1)} \binom{2n-1}{n} z^n \right) \\
&= \frac{1}{2z} \pm \left(\frac{1}{2z} - \sum_{n=1}^{\infty} \frac{1}{(2n-1)} \binom{2n-1}{n} z^{n-1} \right) \\
&= \frac{1}{2z} \pm \left(\frac{1}{2z} - \sum_{n=0}^{\infty} \frac{1}{(2n+1)} \binom{2n+1}{n+1} z^n \right) \\
&= \sum_{n=0}^{\infty} \frac{1}{(2n+1)} \binom{2n+1}{n+1} z^n \\
&= \sum_{n=0}^{\infty} \frac{1}{n+1} \binom{2n}{n} z^n.
\end{aligned}$$

Here we choose minus for the plus-or-minus to get the right answer and then do a little bit of tidying up of the binomial coefficient.

We can check the first few values of $f(n)$:

n	$f(n)$
0	$\binom{0}{0} = 1$
1	$(1/2)\binom{2}{1} = 1$
2	$(1/3)\binom{4}{2} = 6/3 = 2$
3	$(1/4)\binom{6}{3} = 20/4 = 5$

and these are consistent with what we get if we draw all the small binary trees by hand.

The numbers $\frac{1}{n+1}\binom{2n}{n}$ show up in a lot of places in combinatorics, and are known as the **Catalan numbers**.

11.3.10 Summary of operations on generating functions

The following table describes all the nasty things we can do to a generating function. Throughout, we assume $F = \sum f_k z^k$, $G = \sum g_k z^k$, etc.

Operation	Generating functions	Coefficients	Combinatorial interpretation
Find f_0	$f_0 = F(0)$	Returns f_0	Count weight 0 objects.
Find f_k	$f_k = \frac{1}{k!}\frac{d^k}{dz^k}F(z)\|_{z=0}$	Returns f_k	Count weight k objects.
Flatten	$F(1)$	Computes $\sum f_k$	Count all objects, ignoring weights.
Shift right	$G = zF$	$g_k = f_{k-1}$	Add 1 to all weights.
Shift left	$G = z^{-1}(F - F(0))$	$g_k = f_{k+1}$	Subtract 1 from all weights, after removing any weight-0 objects.
Pointing	$G = z\frac{d}{dz}F$	$g_k = kf_k$	A G-thing is an F-thing with a label pointing to one of its units.
Sum	$H = F + G$	$h_k = f_k + g_k$	Disjoint union.
Product	$H = FG$	$h_k = \sum_i f_i g_{k-i}$	Cartesian product.
Composition	$H = F \circ G$	$H = \sum f_k G^k$	To make an H-thing, first choose an F-thing of weight m, then bolt onto it m G-things. The weight of the H-thing is the sum of the weights of the G-things.
Repetition	$G = 1/(1 - F)$	$G = \sum F^k$	A G-thing is a sequence of zero or more F-things. Note: this is just a special case of composition.

11.3.11 Variants

The **exponential generating function** or **egf** for a sequence $a_0, \ldots$ is given by $F(z) = \sum a_n z^n/n!$. For example, the egf for the sequence $1, 1, 1, \ldots$ is $e^z = \sum z^n/n!$. Exponential generating functions admit a slightly different set of operations from ordinary generating functions: differentiation gives left shift (since the factorials compensate for the exponents coming down), multiplying by z gives $b_n = na_{n+1}$, etc. The main application is that the product $F(z)G(z)$ of two egf's gives the sequence whose n-th term is $\sum \binom{n}{k} a_k b_{n-k}$; so for problems where we want that binomial coefficient in the convolution (e.g. when we are building weight n objects not only by choosing a weight-k object plus a weight-$(n-k)$ object but also by arbitrarily rearranging their unit-weight pieces) we want to use an egf rather than an ogf. We won't use these in CS202, but it's worth knowing they exist.

A **probability generating function** or **pgf** is essentially an ordinary generating function where each coefficient a_n is the probability that some random variable equals n. See §12.2 for more details.

11.3.12 Further reading

Rosen [Ros12] discusses some basic facts about generating functions in §8.4. Graham *et al.* [GKP94] give a more thorough introduction. Herbert Wilf's book *generatingfunctionology*, which can be downloaded from the web, will tell you more about the subject than you probably want to know.