Queen Mary
University of London

C50 Enumerative & Asymptotic Combinatorics

Notes 1 **Spring 2003**

This course is about counting. Of course this doesn't mean just counting a single finite set. Usually, we have a family of finite sets indexed by a natural number n, and we want to find $F(n)$, the cardinality of the nth set in the family.

What is counting?

There are several kinds of answer to this question:

- An explicit formula (which may be more or less complicated, and in particular may involve a number of summations).
- A recurrence relation expressing $F(n)$ in terms of values of $F(m)$ for $m < n$.
- A closed form for a *generating function* for F. (The two types of generating function most often used are the *ordinary generating function* $\sum F(n)x^n$, and the *exponential generating function* $\sum F(n)x^n/n!$.) These are elements of the ring $\mathbb{Q}[[x]]$ of *formal power series*. They may or may not converge if a given non-zero complex number is substituted for x. (Formal power series are discussed further in the next section.)

  If a generating function converges, it is possible to find the coefficients by analytic methods (differentiation or contour integration).
- An asymptotic estimate for $F(n)$ is a function $G(n)$, typically expressed in terms of the standard functions of analysis, such that $F(n) - G(n)$ is of smaller order of magnitude than $G(n)$. (If $G(n)$ does not vanish, we can write this as $F(n)/G(n) \to 1$ as $n \to \infty$.) We write $F(n) \sim G(n)$ if this holds. This might be accompanied by an asymptotic estimate for $F(n) - G(n)$, and so on; we obtain an *asymptotic series* for F. (The basics of asymptotic analysis are described further in the third section of this chapter.)

- Related to counting combinatorial objects is the question of generating them. The first thing we might ask for is a system of sequential generation, where we can produce an ordered list of the objects. Again there are two possibilities.

  If the number of objects is $F(n)$, we might ask for a construction which, given i with $0 \leq i \leq F(n)-1$, produces the ith object on the list directly.

  Alternatively, we may simply require a method of moving from each object to the next.

- We could also ask for a method for random generation of an object. If we have a technique for generating the ith object directly, we simply choose a random number in the range $\{0,\ldots,F(n)-1\}$ and generate the corresponding object. If not, we have to rely on other methods such as Markov chains.

Here are a few examples. These will be considered in more detail later in the course.

Example: subsets The number of subsets of $\{1,\ldots,n\}$ is 2^n. Not only is this a simple formula to write down; it is easy to compute as well. At most $2\log_2 n$ integer multiplications are required.

To see this, write n in base 2: $n = 2^{a_1}+2^{a_2}+\cdots+2^{a_r}$, where $a_1 > \cdots > a_r$. Now we can compute 2^{2^i} for $1 \leq i \leq a_1$ by a_1 successive squarings (noting that $2^{2^{i+1}} = \left(2^{2^i}\right)^2$); then $2^n = (2^{2^{a_1}})\cdots(2^{2^{a_r}})$ requires $r-1$ further multiplications.

There is a simple recurrence relation for $F(n) = 2^n$, namely

$$F(0) = 1, \qquad F(n) = 2F(n-1) \text{ for } n \geq 1.$$

Using this, $F(n)$ can be found with just $n-1$ integer doublings.

The ordinary generating function of the sequence (2^n) is $1/(1-2x)$, while the exponential generating function is $\exp(2x)$. (I will use $\exp(x)$ instead of e^x in these notes, except in some places involving calculus.)

No asymptotic estimate is needed, since we have a simple exact formula.

Choosing a random subset, or generating all subsets in order, are easily achieved by the following method. For each $i \in \{0,\ldots,2^n-1\}$, write i in base 2, producing a string of length n of zeros and ones. Now j belongs to the ith subset if and only if the jth symbol in the string is 1.

Example: permutations The number of permutations of $\{1,\ldots,n\}$ is $n!$, defined as usual as the product of the natural numbers from 1 to n. This formula is not so

satisfactory, involving an n-fold product. It can be expressed in other ways, as a sum:

$$n! = \sum_{k=0}^{n} (-1)^{n-k} \binom{n}{k} (n-k)^n,$$

or as an integral:

$$n! = \int_0^\infty x^n \mathrm{e}^{-x} \, \mathrm{d}x.$$

Neither of these is easier to evaluate than the original definition.

The recurrence relation for $F(n) = n!$ is

$$F(0) = 1, \qquad F(n) = nF(n-1) \text{ for } n \geq 1.$$

This leads to the same method of evaluation as we saw earlier.

The ordinary generating function for $F(n) = n!$ fails to converge anywhere. The exponential generating function is $1/(1-x)$, convergent for $|x| < 1$.

As an example to show that convergence is not necessary for a power series to be useful, let

$$\left(1 + \sum_{n \geq 1} n! x^n\right)^{-1} = 1 - \sum_{n \geq 1} c(n) x^n.$$

Then $c(n)$ is the number of connected permutations on $\{1, \ldots, n\}$. (A permutation π is *connected* if there does not exist k with $1 \leq k \leq n-1$ such that π maps $\{1, \ldots, k\}$ to itself.)

An asymptotic estimate for $n!$ is given by *Stirling's formula*:

$$n! \sim \sqrt{2\pi n} \left(\frac{n}{\mathrm{e}}\right)^n.$$

It is possible to generate permutations sequentially, or choose a random permutation, by a method similar to that for subsets.

Example: derangements A derangement is a permutation with no fixed points. Let $d(n)$ be the number of *derangements* of n.

There is a simple formula for $d(n)$: it is the nearest integer to $n!/\mathrm{e}$. This is also satisfactory as an asymptotic expression for $d(n)$; we can supplement it with the fact that $|d(n) - n!/\mathrm{e}| < 1/(n+1)$ for $n > 0$.

This formula is not very good for calculation, since it requires accurate knowledge of e and operations of real (rather than integer) arithmetic. There are, however, two recurrence relations for $d(n)$; the second, especially, leads to efficient calculation:

$$\begin{aligned} d(0) = 1, d(1) = 0, &\qquad d(n) = (n-1)(d(n-1) + d(n-2)) \text{ for } n \geq 2; \\ d(0) = 1, &\qquad d(n) = nd(n-1) + (-1)^n \text{ for } n \geq 1. \end{aligned}$$

The ordinary generating function for $d(n)$ fails to converge, but the exponential generating function is equal to $\exp(-x)/(1-x)$.

Since the probability that a random permutation is a derangement is about $1/\mathrm{e}$, we can choose a random derangement as follows: repeatedly choose a random permutation until a derangement is obtained. The expected number of choices necessary is about e.

Example: partitions The *partition number* $p(n)$ is the number of ways of non-increasing sequences of positive integers with sum n. There is no simple formula for $p(n)$. However, quite a bit is known about it:

- The ordinary generating function is

$$\sum_{n\geq 0} p(n)x^n = \prod_{k\geq 1}(1-x^k)^{-1}.$$

- There is a recurrence relation:

$$p(n) = \sum(-1)^{k-1}p(n-k(3k-1)/2),$$

  where the sum is over all non-zero values of k, positive and negative, for which $n-k(3k-1)/2 \geq 0$. Thus,

$$p(n) = p(n-1)+p(n-2)-p(n-5)-p(n-7)+p(n-12)+\cdots,$$

  where there are about $\sqrt{8n/3}$ terms in the sum.

- The asymptotics of $p(n)$ are rather complicated, and were worked out by Hardy, Littlewood, and Rademacher.

Example: set partitions The *Bell number* $B(n)$ is the number of partitions of the set $\{1,\ldots,n\}$. Again, no simple formula is known, and the asymptotics are very complicated. There is a recurrence relation,

$$B(n) = \sum_{k=1}^{n} \binom{n-1}{k-1} B(n-k),$$

and the exponential generating function is

$$\sum \frac{B(n)x^n}{n!} = \exp(\exp(x)-1).$$

Based on the recurrence one can derive a sequential generation algorithm.

Formal power series

Let R be a commutative ring with identity. A *formal power series* over R is just a function from the natural numbers to R; that is, an infinite sequence

$$r_0, r_1, r_2, \ldots, r_n, \ldots \tag{1}$$

of elements of R. We define addition and multiplication of such infinite series to make the set of formal power series into a ring. The definitions look more natural if we write the sequence (1) as

$$r_0 + r_1 x + r_2 x^2 + \cdots + r_n x^n + \cdots \tag{2}$$

The symbol x in this expression is just a dummy with no meaning; the "power" of x allows us to keep track of our place in the series. No infinite summation is actually involved! We denote the set of all formal power series by $R[[x]]$. If we had used a different symbol, say y, in the expression (2), we would write $R[[y]]$ instead. We often abbreviate (2) to

$$\sum_{n\geq 0} r_n x^n. \tag{3}$$

A *polynomial* is simply a formal power series in which all but finitely many of the terms are zero. The *degree* of a polynomial is the index of the last non-zero term. The set of polynomials is denoted by $R[x]$.

We define addition and multiplication of formal power series by

$$\begin{aligned}\left(\sum_{n\geq 0} r_n x^n\right) + \left(\sum_{n\geq 0} s_n x^n\right) &= \sum_{n\geq 0} (r_n + s_n) x^n, \\ \left(\sum_{n\geq 0} r_n x^n\right) \cdot \left(\sum_{n\geq 0} s_n x^n\right) &= \sum_{n\geq 0} t_n x^n,\end{aligned}$$

where

$$t_n = \sum_{k=0}^{n} r_k s_{n-k}.$$

Note that these operations involve only finite additions and multiplications of ring elements.

With these operations, $R[[x]]$ is a ring, and $R[x]$ a subring. We don't stop to prove this, as the verifications are routine.

Various other apparently "infinitary" operations can be defined which only involve finite sums and products. For example,

- Suppose that $f_0, f_1, \ldots \in R[[x]]$ have the property that the index of the smallest non-zero term in f_n tends to infinity with n. Then

$$\sum_{n\geq 0} f_n$$

is defined. In particular, if $f_n = r_n x^n$, the condition is satisfied, and this definition of the infinite sum agrees with our notation for the formal power series $\sum r_n x^n$.

- With the same conditions,

$$\prod_{n\geq 0}(1+f_n)$$

is defined: it is the sum of terms, each of which is the product of finitely many f_n (taking 1 from the remaining factors in the infinite product); and by assumption only finitely many such products contribute to the coefficient of x^n for any n.

- Let f and g be formal power series in which the constant term of g is zero. Then the result of substituting g into f is defined: if $f(x) = \sum r_n x^n$, then $f(g(x)) = \sum r_n g^n$.

A result which is important for enumeration is the following, though we are more concerned with the method of proof than the statement.

Proposition 1 *A formal power series is invertible if and only if its constant term is invertible.*

Proof Suppose that $f = \sum r_n x^n$ and $g = \sum s_n x^n$ satisfy $fg = 1$. Considering the term of degree zero, we see that $r_0 s_0 = 1$, so that r_0 is invertible.

Conversely, suppose that $r_0 s_0 = 1$, where $f = \sum r_n x^n$. The inverse $g = \sum s_n x^n$ must satisfy

$$\sum_{k=0}^{n} r_k s_{n-k} = 0$$

for $n > 0$; so

$$s_n = -s_0 \sum_{k=1}^{n} r_k s_{n-k}.$$

Thus the coefficients of g satisfy a linear recurrence relation, and can be determined recursively.

In general, knowledge of the inverse of a formal power series is equivalent to knowledge of a linear recurrence relation for its coefficients.

Example: Fibonacci numbers Let $f(x) = 1 - x - x^2$. Then the coefficients of the inverse $(1 - x - x^2)^{-1} = \sum s_n x^n$ satisfy the recurrence

$$s_0 = s_1 = 1, \qquad s_n = s_{n-1} + s_{n-2} \text{ for } n \geq 2;$$

in other words, they are the Fibonacci numbers.

For the purposes of enumeration, the coefficients of formal power series are usually integers or rational numbers. Often it is convenient to consider them as real numbers, and apply to them the processes of analysis.

For example, considering the Fibonacci numbers above, let α and β be the roots of the quadratic equation $z^2 - z - 1 = 0$: thus, $\alpha = (\sqrt{5}+1)/2$ and $\beta = (-\sqrt{5}+1)/2$. Then

$$\begin{aligned} \frac{1}{1-x-x^2} &= \frac{1}{\alpha-\beta}\left(\frac{\alpha}{1-\alpha x} - \frac{\beta}{1-\beta x}\right) \\ &= \frac{1}{\sqrt{5}}\left(\sum_{n\geq 0} \alpha^{n+1}x^n - \sum_{n\geq 0} \beta^{n+1}x^n\right); \end{aligned}$$

so the nth Fibonacci number is

$$F_n = \frac{1}{\sqrt{5}}(\alpha^{n+1} - \beta^{n+1}).$$

Since $|\beta| < 1$, we see that F_n is the nearest integer to $\alpha^{n+1}/\sqrt{5}$.

Particular formal power series of great importance include

$$\begin{aligned} \exp(x) &= \sum_{n\geq 0} \frac{x^n}{n!}, \\ \log(1+x) &= \sum_{n\geq 1} \frac{(-1)^{n-1}x^n}{n}. \end{aligned}$$

Asymptotics

We introduce the notation for describing the asymptotic behaviour of functions here, though we will not do any serious asymptotic estimation for a while.

Let F and G be functions of the natural number n. For convenience we assume that G does not vanish. We write

- $F = O(G)$ if $F(n)/G(n)$ is bounded above as $n \to \infty$;

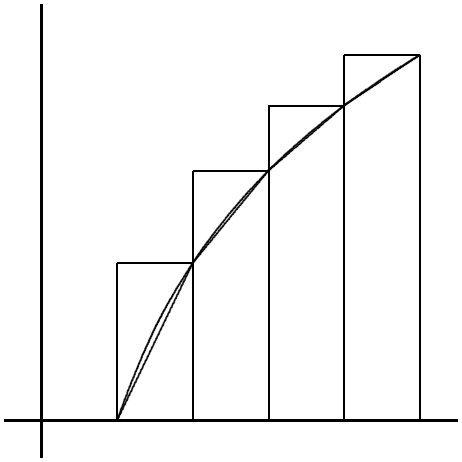

Figure 1: Stirling's formula

- $F = \Omega(G)$ if $F(n)/G(n)$ is bounded below as $n \to \infty$;
- $F = o(G)$ if $F(n)/G(n) \to 0$ as $n \to \infty$;
- $F \sim G$ if $F/G \to 1$ as $n \to \infty$.

Typically, F is a combinatorial enumeration function, and G a combination of standard functions of analysis. For example, Stirling's formula gives the asymptotics of the number of permutations of $\{1,\ldots,n\}$. We give the proof as an illustration.

Theorem 2

$$n! \sim \sqrt{2\pi n}\left(\frac{n}{\mathrm{e}}\right)^n$$

Proof We outline a proof, apart from identifying the constant $\sqrt{2\pi}$. Consider the graph of the function $y = \log x$ between $x = 1$ and $x = n$, together with the piecewise linear functions shown in Figure 1.

Let $f(x) = \log x$, let $g(x)$ be the function whose value is $\log m$ for $m \le x < m+1$, and let $h(x)$ be the function defined by the polygon with vertices $(m, \log m)$, for $1 \le m \le n$. Clearly

$$\int_1^n g(x)\,\mathrm{d}x = \log 2 + \cdots + \log n = \log n!\,.$$

The difference between the integrals of g and h is the sum of the areas of triangles with base 1 and total height $\log n$; that is, $\frac{1}{2}\log n$.

Some calculus shows that the difference between the integrals of f and g tends to a finite limit c as $n \to \infty$. (Show that $|f(x) - g(x)| = O(1/m^2)$ for $m \leq x < m+1$.)

Finally, a simple integration shows that

$$\int_1^n f(x)\,\mathrm{d}x = n\log n - n + 1.$$

We conclude that

$$\log n! = n\log n - n + \tfrac{1}{2}\log n + (1-c) + o(1),$$

from which the result follows.

The series $G_0(n) + G_1(n) + G_2(n) + \cdots$ is an *asymptotic series* for $F(n)$ if

$$F(n) - \sum_{j=0}^{i-1} G_j(n) \sim G_i(n)$$

for $i \geq 0$. (So in particular $F(n) \sim G_0(n)$, $F(n) - G_0(n) \sim G_1(n)$, and so on. Note that $G_i(n) = o(G_{i-1}(n))$ for all i.)

Warnings:

- an asymptotic series is not necessarily convergent;
- it is not necessarily the case that taking more terms in the series gives a better approximation to $F(n)$ for a fixed n.

For example, Stirling's formula can be extended to an asymptotic series for $n!$, namely

$$\sqrt{2\pi n}\left(\frac{n}{\mathrm{e}}\right)^n \left(1 + \frac{1}{12n} + \frac{1}{288n^2} + \cdots\right).$$

Regarding a generating function for a sequence as a function of a real or complex variable is a powerful method for studying the asymptotic behaviour of the sequence. We will see examples of this later; here is a simple observation.

Suppose that $A(z) = \sum a_n z^n$ defines a function which is analytic in some neighbourhood of the origin in the complex plane. Suppose that the smallest modulus of a singularity of $A(z)$ is R. Then $\limsup a_n^{1/n} = 1/R$, so a_n is bounded by $(c+\varepsilon)^n$ but not by $(c-\varepsilon)^n$ for large n, where $c = 1/R$.

For example, we saw that the generating function of the Fibonacci numbers is $1/(1-z-z^2)$. So these numbers grow roughly like α^n, where α is the reciprocal of the smaller root of $1-z-z^2 = 0$, namely $\alpha = (1+\sqrt{5})/2$.

On the other hand, if $A(z)$ is analytic everywhere, then $a_n \le \varepsilon^n$ for $n > n_0(\varepsilon)$, for any positive ε. Indeed, $a_n = o(\varepsilon^n)$ for any positive ε.

For example, if $B(n)$ is the nth Bell number, then

$$\sum_{n\ge 0} \frac{B(n)z^n}{n!} = \mathrm{e}^{\mathrm{e}^z-1},$$

which is analytic everywhere. So $B(n) = o(\varepsilon^n n!)$, for any positive ε.

Complexity

A formula like 2^n (the number of subsets of an n-set) can be evaluated quickly for a given value of n. A more complicated formula with multiple sums and products will take longer to calculate. We could regard a formula which takes more time to evaluate than it would take to generate all the objects and count them as being useless in practice, even if it has theoretical value.

Traditional computational complexity theory refers to decision problems, where the answer is just "yes" or "no" (for example, "Does this graph have a Hamiltonian circuit?"). The size of an instance of a problem is measured by the number of bits of data required to specify the problem (for example, $n(n-1)/2$ bits to specify a graph on n vertices). Then the time complexity of a problem is the function f, where $f(n)$ is the maximum number of steps required by a Turing machine to compute the answer for an instance of size n. To allow for variations in the format of the input data and in the exact specification of a Turing machine, complexity classes are defined with a broad brush: for example, P (or "polynomial-time") consists of all problems whose time complexity is at most n^c for some constant c. (For more details, see Garey and Johnson, *Computers and Intractability*.)

For counting problems, the answer is a number rather than a single Boolean value (for example, "How many Hamiltonian circuits does this graph have?"). Complexity theorists have defined the complexity class #P ("number-P") for this purpose.

Even this class is not really appropriate for counting problems of the type we mostly consider. Consider, for example, the question "How many partitions does an n-set have?" The input data is the integer n, which (if written in base 2) requires only $m = \lceil 1 + \log_2 n \rceil$ bits to specify. The question asks us to calculate the Bell number $B(n)$, which is greater than 2^{n-1} for $n > 2$, and so it takes time exponential in m simply to write down the answer! To get round this difficulty, it is usual to pretend that the size of the input data is actually n rather than $\log n$. (We can imagine that n is given by writing n consecutive 1s on the input tape of the Turing machine, that is, by writing n as a tally rather than in base 2.)

We have seen that computing 2^n (the number of subsets of an n-set) requires only $O(\log n)$ integer multiplications. But the integers may have as many as n digits, so each multiplication takes $O(n)$ Turing machine steps. Similarly, the solution to a recurrence relation can be computed in time polynomial in n, provided that each individual computation can be.

On the other hand, a method which involves generating and testing every subset or permutation will take exponentially long, even if the generation and testing can be done efficiently.

A notion of complexity relevant to this situation is the polynomial delay model, which asks that the time required to generate each object should be at most n^c for some fixed c, even if the number of objects to be generated is greater than polynomial.

Of course, it is easy to produce combinatorial problems whose solution grows faster than, say, the exponential of a polynomial. For example, how many intersecting families of subsets of an n-set are there? The total number, for n odd, lies between $2^{2^{n-1}}$ and 2^{2^n}, so that even writing down the answer takes time exponential in n.

We will not consider complexity questions further in this course.