An Introduction to Mathematical Methods in Combinatorics

Renzo Sprugnoli
Dipartimento di Sistemi e Informatica
Viale Morgagni, 65 - Firenze (Italy)

January 18, 2006

Contents

Chapter 1

Introduction

1.1 What is the Analysis of an Algorithm

An *algorithm* is a finite sequence of unambiguous rules for solving a problem. Once the algorithm is started with a particular *input*, it will always end, obtaining the correct answer or *output*, which is the solution to the given problem. An algorithm is realized on a computer by means of a *program*, i.e., a set of instructions which cause the computer to perform the elaborations intended by the algorithm.

So, an algorithm is independent of any computer, and, in fact, the word was used for a long time before computers were invented. Leonardo Fibonacci called "algorisms" the rules for performing the four basic operations (addition, subtraction, multiplication and division) with the newly introduced arabic digits and the positional notation for numbers. "Euclid's algorithm" for evaluating the greatest common divisor of two integer numbers was also well known before the appearance of computers.

Many algorithms can exist which solve the same problem. Some can be very skillful, others can be very simple and straight-forward. A natural problem, in these cases, is to choose, if any, the best algorithm, in order to realize it as a program on a particular computer. Simple algorithms are more easily programmed, but they can be very slow or require large amounts of computer memory. The problem is therefore to have some means to judge the speed and the quantity of computer resources a given algorithm uses. The aim of the "Analysis of Algorithms" is just to give the mathematical bases for describing the behavior of algorithms, thus obtaining exact criteria to compare different algorithms performing the same task, or to see if an algorithm is sufficiently good to be efficiently implemented on a computer.

Let us consider, as a simple example, the *problem of searching*. Let S be a given set. In practical cases S is the set $\mathbb{N}$ of natural numbers, or the set $\mathbb{Z}$ of integer numbers, or the set $\mathbb{R}$ of real numbers or also the set A^* of the words on some alphabet A. However, as a matter of fact, the nature of S is not essential, because we always deal with a suitable binary representation of the elements in S on a computer, and have therefore to be considered as "words" in the computer memory. The "problem of searching" is as follows: we are given a finite ordered subset $T = (a_1, a_2, \ldots, a_n)$ of S (usually called a *table*, its elements referred to as *keys*), an element $s \in S$ and we wish to know whether $s \in T$ or $s \notin T$, and in the former case which element a_k in T it is.

Although the mathematical problem "$s \in T$ or not" has almost no relevance, the searching problem is basic in computer science and many algorithms have been devised to make the process of searching as fast as possible. Surely, the most straight-forward algorithm is *sequential searching*: we begin by comparing s and a_1 and we are finished if they are equal. Otherwise, we compare s and a_2, and so on, until we find an element $a_k = s$ or reach the end of T. In the former case the search is *successful* and we have determined the element in T equal to s. In the latter case we are convinced that $s \notin T$ and the search is *unsuccessful*.

The analysis of this (simple) algorithm consists in finding one or more mathematical expressions describing in some way the number of operations performed by the algorithm as a function of the number n of the elements in T. This definition is intentionally vague and the following points should be noted:

- an algorithm can present several aspects and therefore may require several mathematical expressions to be fully understood. For example, for what concerns the sequential search algorithm, we are interested in what happens during a successful or unsuccessful search. Besides, for a successful search, we wish to know what the worst case is (*Worst Case Analysis*) and what the average case is (*Average Case Analysis*) with respect to all the tables containing n elements or, for a fixed table, with respect to the n elements it contains;

- the operations performed by an algorithm can be

of many different kinds. In the example above the only operations involved in the algorithm are comparisons, so no doubt is possible. In other algorithms we can have arithmetic or logical operations. Sometimes we can also consider more complex operations as square roots, list concatenation or reversal of words. Operations depend on the nature of the algorithm, and we can decide to consider as an "operation" also very complicated manipulations (e.g., extracting a random number or performing the differentiation of a polynomial of a given degree). The important point is that every instance of the "operation" takes on about the same time or has the same complexity. If this is not the case, we can give different weights to different operations or to different instances of the same operation.

We observe explicitly that we never consider execution time as a possible parameter for the behavior of an algorithm. As stated before, algorithms are independent of any particular computer and should never be confused with the programs realizing them. Algorithms are only related to the "logic" used to solve the problem; programs can depend on the ability of the programmer or on the characteristics of the computer.

1.2 The Analysis of Sequential Searching

The analysis of the sequential searching algorithm is very simple. For a successful search, the worst case analysis is immediate, since we have to perform n comparisons if $s = a_n$ is the last element in T. More interesting is the average case analysis, which introduces the first mathematical device of algorithm analysis. To find the average number of comparisons in a successful search we should sum the number of comparisons necessary to find any element in T, and then divide by n. It is clear that if $s = a_1$ we only need a single comparison; if $s = a_2$ we need two comparisons, and so on. Consequently, we have:

$$C_n = \frac{1}{n}\left(1 + 2 + \cdots + n\right) = \frac{1}{n}\frac{n(n+1)}{2} = \frac{n+1}{2}$$

This result is intuitively clear but important, since it shows in mathematical terms that the number of comparisons performed by the algorithm (and hence the time taken on a computer) grows linearly with the dimension of T.

The concluding step of our analysis was the execution of a sum—a well-known sum in the present case. This is typical of many algorithms and, as a matter of fact, the ability in performing sums is an important technical point for the analyst. A large part of our efforts will be dedicated to this topic.

Let us now consider an unsuccessful search, for which we only have an Average Case analysis. If U_k denotes the number of comparisons necessary for a table with k elements, we can determine U_n in the following way. We compare s with the first element a_1 in T and obviously we find $s \neq a_1$, so we should go on with the table $T' = T \setminus \{a_1\}$, which contains $n-1$ elements. Consequently, we have:

$$U_n = 1 + U_{n-1}$$

This is a *recurrence relation*, that is an expression relating the value of U_n with other values of U_k having $k < n$. It is clear that if some value, e.g., U_0 or U_1 is known, then it is possible to find the value of U_n, for every $n \in \mathbb{N}$. In our case, U_0 is the number of comparisons necessary to find out that an element s does not belong to a table containing no element. Hence we have the *initial condition* $U_0 = 0$ and we can *unfold* the preceding recurrence relation:

$$\begin{aligned} U_n &= 1 + U_{n-1} = 1 + 1 + U_{n-2} = \cdots = \\ &= \underbrace{1 + 1 + \cdots + 1}_{n \text{ times}} + U_0 = n \end{aligned}$$

Recurrence relations are the other mathematical device arising in algorithm analysis. In our example the recurrence is easily transformed into a sum, but as we shall see this is not always the case. In general we have the problem of *solving* a recurrence, i.e., to find an explicit expression for U_n, starting with the recurrence relation and the initial conditions. So, another large part of our efforts will be dedicated to the solution or recurrences.

1.3 Binary Searching

Another simple example of analysis can be performed with the binary search algorithm. Let S be a given ordered set. The ordering must be total, as the numerical order in $\mathbb{N}$, $\mathbb{Z}$ or $\mathbb{R}$ or the lexicographical order in A^*. If $T = (a_1, a_2, \ldots, a_n)$ is a finite ordered subset of S, i.e., a table, we can always imagine that $a_1 < a_2 < \cdots < a_n$ and consider the following algorithm, called *binary searching*, to search for an element $s \in S$ in T. Let a_i the median element in T, i.e., $i = \lfloor (n+1)/2 \rfloor$, and compare it with s. If $s = a_i$ then the search is successful; otherwise, if $s < a_i$, perform the same algorithm on the subtable $T' = (a_1, a_2, \ldots, a_{i-1})$; if instead $s > a_i$ perform the same algorithm on the subtable $T'' = (a_{i+1}, a_{i+2}, \ldots, a_n)$. If at any moment the table on which we perform the search is reduced to the empty set $\emptyset$, then the search is unsuccessful.

Let us consider first the Worst Case analysis of this algorithm. For a successful search, the element s is only found at the last step of the algorithm, i.e., when the subtable on which we search is reduced to a single element. If B_n is the number of comparisons necessary to find s in a table T with n elements, we have the recurrence:

$$B_n = 1 + B_{\lfloor n/2 \rfloor}$$

In fact, we observe that every step reduces the table to $\lfloor n/2 \rfloor$ or to $\lfloor (n-1)/2 \rfloor$ elements. Since we are performing a Worst Case analysis, we consider the worse situation. The initial condition is $B_1 = 1$, relative to the table (s), to which we should always reduce. The recurrence is not so simple as in the case of sequential searching, but we can simplify everything considering a value of n of the form $2^k - 1$. In fact, in such a case, we have $\lfloor n/2 \rfloor = \lfloor (n-1)/2 \rfloor = 2^{k-1} - 1$, and the recurrence takes on the form:

$$B_{2^k-1} = 1 + B_{2^{k-1}-1} \quad \text{or} \quad \beta_k = 1 + \beta_{k-1}$$

if we write β_k for B_{2^k-1}. As before, unfolding yields $\beta_k = k$, and returning to the B's we find:

$$B_n = \log_2(n+1)$$

by our definition $n = 2^k - 1$. This is valid for every n of the form $2^k - 1$ and for the other values this is an approximation, a rather good approximation, indeed, because of the very slow growth of logarithms.

We observe explicitly that for $n = 1,000,000$, a sequential search requires about 500,000 comparisons on the average for a successful search, whereas binary searching only requires $\log_2(1,000,000) \approx 20$ comparisons. This accounts for the dramatic improvement that binary searching operates on sequential searching, and the analysis of algorithms provides a mathematical proof of such a fact.

The Average Case analysis for successful searches can be accomplished in the following way. There is only one element that can be found with a single comparison: the median element in T. There are two elements that can be found with two comparisons: the median elements in T' and in T''. Continuing in the same way we find the average number A_n of comparisons as:

$$\begin{aligned} A_n &= \frac{1}{n}\,(1 + 2 + 2 + 3 + 3 + 3 + 3 + 4 + \cdots \\ &\quad \cdots + (1 + \lfloor \log_2(n) \rfloor)) \end{aligned}$$

The value of this sum can be found explicitly, but the method is rather difficult and we delay it until later (see Section 4.7). When $n = 2^k - 1$ the expression simplifies:

$$A_{2^k-1} = \frac{1}{2^k - 1} \sum_{j=1}^{k} j 2^{j-1} = \frac{k2^k - 2^k + 1}{2^k - 1}$$

This sum also is not immediate, but the reader can check it by using mathematical induction. If we now write $k2^k - 2^k + 1 = k(2^k - 1) + k - (2^k - 1)$, we find:

$$A_n = k + \frac{k}{n} - 1 = \log_2(n+1) - 1 + \frac{\log_2(n+1)}{n}$$

which is only a little better than the worst case.

For unsuccessful searches, the analysis is now very simple, since we have to proceed as in the Worst Case analysis and at the last comparison we have a failure instead of a success. Consequently, $U_n = B_n$.

1.4 Closed Forms

The sign "=" between two numerical expressions denotes their numerical equivalence as for example:

$$\sum_{k=0}^{n} k = \frac{n(n+1)}{2}$$

Although algebraically or numerically equivalent, two expressions can be computationally quite different. In the example, the left-hand expression requires n sums to be evaluated, whereas the right-hand expression only requires a sum, a multiplication and a halving. For n also moderately large (say $n \geq 5$) nobody would prefer computing the left-hand expression rather than the right-hand one. A computer evaluates this latter expression in a few nanoseconds, but can require some milliseconds to compute the former, if only n is greater than 10,000. The important point is that the evaluation of the right-hand expression is independent of n, whilst the left-hand expression requires a number of operations growing linearly with n.

A *closed form expression* is an expression, depending on some parameter n, the evaluation of which does not require a number of operations depending on n. Another example we have already found is:

$$\sum_{k=0}^{n} k 2^{k-1} = n2^n - 2^n + 1 = (n-1)2^n + 1$$

Again, the left-hand expression is not in closed form, whereas the right-hand one is. We observe that $2^n = 2 \times 2 \times \cdots \times 2$ (n times) seems to require $n-1$ multiplications. In fact, however, 2^n is a simple shift in a binary computer and, more in general, every power $\alpha^n = \exp(n \ln \alpha)$ can be always computed with the maximal accuracy allowed by a computer in constant time, i.e., in a time independent of α and n. This is because the two elementary functions $\exp(x)$ and $\ln(x)$ have the nice property that their evaluation is independent of their argument. The same property holds true for the most common numerical functions,

as the trigonometric and hyperbolic functions, the Γ and ψ functions (see below), and so on.

As we shall see, in algorithm analysis there appear many kinds of "special" numbers. Most of them can be reduced to the computation of some basic quantities, which are considered to be in closed form, although apparently they depend on some parameter n. The three main quantities of this kind are the *factorial*, the *harmonic numbers* and the *binomial coefficients*. In order to justify the previous sentence, let us anticipate some definitions, which will be discussed in the next chapter, and give a more precise presentation of the Γ and ψ functions.

The Γ-*function* is defined by a definite integral:

$$\Gamma(x) = \int_0^\infty t^{x-1} e^{-t}\, dt.$$

By integrating by parts, we obtain:

$$\begin{aligned}\Gamma(x+1) &= \int_0^\infty t^x e^{-t}\, dt = \\ &= \left[-t^x e^{-t}\right]_0^\infty + \int_0^\infty x t^{x-1} e^{-t}\, dt \\ &= x\Gamma(x)\end{aligned}$$

which is a basic, recurrence property of the Γ-function. It allows us to reduce the computation of $\Gamma(x)$ to the case $1 \le x \le 2$. In this interval we can use a polynomial approximation:

$$\Gamma(x+1) = 1 + b_1 x + b_2 x^2 + \cdots + b_8 x^8 + \epsilon(x)$$

where:

$$\begin{array}{rclrcl} b_1 &=& -0.577191652 & b_5 &=& -0.756704078 \\ b_2 &=& 0.988205891 & b_6 &=& 0.482199394 \\ b_3 &=& -0.897056937 & b_7 &=& -0.193527818 \\ b_4 &=& 0.918206857 & b_8 &=& 0.035868343 \end{array}$$

The error is $|\epsilon(x)| \le 3 \times 10^{-7}$. Another method is to use Stirling's approximation:

$$\begin{aligned}\Gamma(x) &= e^{-x} x^{x-0.5} \sqrt{2\pi} \left(1 + \frac{1}{12x} + \frac{1}{288x^2} - \right. \\ &\quad \left. - \frac{139}{51840x^3} - \frac{571}{2488320x^4} + \cdots\right).\end{aligned}$$

Some special values of the Γ-function are directly obtained from the definition. For example, when $x = 1$ the integral simplifies and we immediately find $\Gamma(1) = 1$. When $x = 1/2$ the definition implies:

$$\Gamma(1/2) = \int_0^\infty t^{1/2-1} e^{-t}\, dt = \int_0^\infty \frac{e^{-t}}{\sqrt{t}}\, dt.$$

By performing the substitution $y = \sqrt{t}$ ($t = y^2$ and $dt = 2ydy$), we have:

$$\Gamma(1/2) = \int_0^\infty \frac{e^{-y^2}}{y} 2y dy = 2\int_0^\infty e^{-y^2} dy = \sqrt{\pi}$$

the last integral being the famous *Gauss' integral*. Finally, from the recurrence relation we obtain:

$$\Gamma(1/2) = \Gamma(1 - 1/2) = -\frac{1}{2}\Gamma(-1/2)$$

and therefore:

$$\Gamma(-1/2) = -2\Gamma(1/2) = -2\sqrt{\pi}.$$

The Γ function is defined for every $x \in \mathbb{C}$, except when x is a negative integer, where the function goes to infinity; the following approximation can be important:

$$\Gamma(-n + \epsilon) \approx \frac{(-1)^n}{n!}\, \frac{1}{\epsilon}.$$

When we unfold the basic recurrence of the Γ-function for $x = n$ an integer, we find $\Gamma(n+1) = n \times (n-1) \times \cdots \times 2 \times 1$. The factorial $\Gamma(n+1) = n! = 1 \times 2 \times 3 \times \cdots \times n$ seems to require $n - 2$ multiplications. However, for n large it can be computed by means of the *Stirling's formula*, which is obtained from the same formula for the Γ-function:

$$\begin{aligned} n! &= \Gamma(n+1) = n\Gamma(n) = \\ &= \sqrt{2\pi n} \left(\frac{n}{e}\right)^n \left(1 + \frac{1}{12n} + \frac{1}{288n^2} + \cdots\right). \end{aligned}$$

This requires only a fixed amount of operations to reach the desired accuracy.

The function $\psi(x)$, called ψ-*function* or *digamma function*, is defined as the logarithmic derivative of the Γ-function:

$$\psi(x) = \frac{d}{dx} \ln \Gamma(x) = \frac{\Gamma'(x)}{\Gamma(x)}.$$

Obviously, we have:

$$\begin{aligned}\psi(x+1) &= \frac{\Gamma'(x+1)}{\Gamma(x+1)} = \frac{\frac{d}{dx} x\Gamma(x)}{x\Gamma(x)} = \\ &= \frac{\Gamma(x) + x\Gamma'(x)}{x\Gamma(x)} = \frac{1}{x} + \psi(x)\end{aligned}$$

and this is a basic property of the digamma function. By this formula we can always reduce the computation of $\psi(x)$ to the case $1 \le x \le 2$, where we can use the approximation:

$$\psi(x) = \ln x - \frac{1}{2x} - \frac{1}{12x^2} + \frac{1}{120x^4} - \frac{1}{252x^6} + \cdots.$$

By the previous recurrence, we see that the digamma function is related to the harmonic numbers $H_n = 1 + 1/2 + 1/3 + \cdots + 1/n$. In fact, we have:

$$H_n = \psi(n+1) + \gamma$$

where $\gamma = 0.57721566\ldots$ is the *Mascheroni-Euler constant*. By using the approximation for $\psi(x)$, we

obtain an approximate formula for the Harmonic numbers:

$$H_n = \ln n + \gamma + \frac{1}{2n} - \frac{1}{12n^2} + \cdots$$

which shows that the computation of H_n does not require $n-1$ sums and $n-1$ inversions as it can appear from its definition.

Finally, the binomial coefficient:

$$\begin{aligned}\binom{n}{k} &= \frac{n(n-1)\cdots(n-k+1)}{k!} = \\ &= \frac{n!}{k!(n-k)!} = \\ &= \frac{\Gamma(n+1)}{\Gamma(k+1)\Gamma(n-k+1)}\end{aligned}$$

can be reduced to the computation of the Γ function or can be approximated by using the Stirling's formula for factorials. The two methods are indeed the same. We observe explicitly that the last expression shows that binomial coefficients can be defined for every $n, k \in \mathbb{C}$, except that k cannot be a negative integer number.

The reader can, as a very useful exercise, write computer programs to realize the various functions mentioned in the present section.

1.5 The Landau notation

To the mathematician Edmund Landau is ascribed a special notation to describe the general behavior of a function $f(x)$ when x approaches some definite value. We are mainly interested to the case $x \to \infty$, but this should not be considered a restriction. Landau notation is also known as *O-notation* (or big-oh notation), because of the use of the letter O to denote the desired behavior.

Let us consider functions $f : \mathbb{N} \to \mathbb{R}$ (i.e., sequences of real numbers); given two functions $f(n)$ and $g(n)$, we say that $f(n)$ is $O(g(n))$, or that $f(n)$ *is in the order of* $g(n)$, if and only if:

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} < \infty$$

In formulas we write $f(n) = O(g(n))$ or also $f(n) \sim g(n)$. Besides, if we have at the same time:

$$\lim_{n\to\infty} \frac{g(n)}{f(n)} < \infty$$

we say that $f(n)$ *is in the same order as* $g(n)$ and write $f(n) = \Theta(g(n))$ or $f(n) \approx g(n)$.

It is easy to see that "$\sim$" is an order relation between functions $f : \mathbb{N} \to \mathbb{R}$, and that "$\approx$" is an equivalence relation. We observe explicitly that when $f(n)$ is in the same order as $g(n)$, a constant $K \neq 0$ exists such that:

$$\lim_{n\to\infty} \frac{f(n)}{Kg(n)} = 1 \qquad \text{or} \qquad \lim_{n\to\infty} \frac{f(n)}{g(n)} = K;$$

the constant K is very important and will often be used.

Before making some important comments on Landau notation, we wish to introduce a last definition: we say that $f(n)$ *is of smaller order than* $g(n)$ and write $f(n) = o(g(n))$, iff:

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0.$$

Obviously, this is in accordance with the previous definitions, but the notation introduced (the *small-oh notation*) is used rather frequently and should be known.

If $f(n)$ and $g(n)$ describe the behavior of two algorithms A and B solving the same problem, we will say that A *is asymptotically better* than B iff $f(n) = o(g(n))$; instead, the two algorithms are *asymptotically equivalent* iff $f(n) = \Theta(g(n))$. This is rather clear, because when $f(n) = o(g(n))$ the number of operations performed by A is substantially less than the number of operations performed by B. However, when $f(n) = \Theta(g(n))$, the number of operations is the same, except for a constant quantity K, which remains the same as $n \to \infty$. The constant K can simply depend on the particular realization of the algorithms A and B, and with two different implementations we may have $K < 1$ or $K > 1$. Therefore, in general, when $f(n) = \Theta(g(n))$ we cannot say which algorithm is better, this depending on the particular realization or on the particular computer on which the algorithms are run. Obviously, if A and B are both realized on the same computer and in the best possible way, a value $K < 1$ tells us that algorithm A is relatively better than B, and vice versa when $K > 1$.

It is also possible to give an absolute evaluation for the performance of a given algorithm A, whose behavior is described by a sequence of values $f(n)$. This is done by comparing $f(n)$ against an *absolute scale* of values. The scale most frequently used contains powers of n, logarithms and exponentials:

$$\begin{aligned}O(1) &< O(\ln n) < O(\sqrt{n}) < O(n) < \cdots \\ &< O(n\ln n) < O(n\sqrt{n}) < O(n^2) < \cdots \\ &< O(n^5) < \cdots < O(e^n) < \cdots \\ &< O(e^{e^n}) < \cdots.\end{aligned}$$

This scale reflects well-known properties: the logarithm grows more slowly than any power n^ϵ, however small ϵ, while e^n grows faster than any power

n^k, however large k, independent of n. Note that $n^n = e^{n \ln n}$ and therefore $O(e^n) < O(n^n)$. As a matter of fact, the scale is not complete, as we obviously have $O(n^{0.4}) < O(\sqrt{n})$ and $O(\ln \ln n) < O(\ln n)$, but the reader can easily fill in other values, which can be of interest to him or her.

We can compare $f(n)$ to the elements of the scale and decide, for example, that $f(n) = O(n)$; this means that algorithm A performs at least as well as any algorithm B whose behavior is described by a function $g(n) = \Theta(n)$.

Let $f(n)$ be a sequence describing the behavior of an algorithm A. If $f(n) = O(1)$, then the algorithm A performs in a constant time, i.e., in a time independent of n, the parameter used to evaluate the algorithm behavior. Algorithms of this type are the best possible algorithms, and they are especially interesting when n becomes very large. If $f(n) = O(n)$, the algorithm A is said to be *linear*; if $f(n) = O(\ln n)$, A is said to be *logarithmic*; if $f(n) = O(n^2)$, A is said to be *quadratic*; and if $f(n) \geq O(e^n)$, A is said to be *exponential*. In general, if $f(n) \leq O(n^r)$ for some $r \in \mathbb{R}$, then A is said to be *polynomial*.

As a standard terminology, mainly used in the "Theory of Complexity", polynomial algorithms are also called *tractable*, while exponential algorithms are called *intractable*. These names are due to the following observation. Suppose we have a linear algorithm A and an exponential algorithm B, not necessarily solving the same problem. Also suppose that an hour of computer time executes both A and B with $n = 40$. If a new computer is used which is 1000 times faster than the old one, in an hour algorithm A will execute with m such that $f(m) = K_A m = 1000 K_A n$, or $m = 1000n = 40,000$. Therefore, the problem solved by the new computer is 1000 times larger than the problem solved by the old computer. For algorithm B we have $g(m) = K_B e^m = 1000 K_B e^n$; by simplifying and taking logarithms, we find $m = n + \ln 1000 \approx n + 6.9 < 47$. Therefore, the improvement achieved by the new computer is almost negligible.

Chapter 2

Special numbers

A *sequence* is a mapping from the set $\mathbb{N}$ of natural numbers into some other set of numbers. If $f : \mathbb{N} \to \mathbb{R}$, the sequence is called a sequence of real numbers; if $f : \mathbb{N} \to \mathbb{Q}$, the sequence is called a sequence of rational numbers; and so on. Usually, the image of a $k \in \mathbb{N}$ is denoted by f_k instead of the traditional $f(k)$, and the whole sequence is abbreviated as $(f_0, f_1, f_2, \ldots) = (f_k)_{k \in N}$. Because of this notation, an element $k \in \mathbb{N}$ is called an *index*.

Often, we also study *double sequences*, i.e., mappings $f : \mathbb{N} \times \mathbb{N} \to \mathbb{R}$ or some other numeric set. In this case also, instead of writing $f(n, k)$ we will usually write $f_{n,k}$ and the whole sequence will be denoted by $\{f_{n,k} \,|\, n, k \in \mathbb{N}\}$ or $(f_{n,k})_{n,k \in N}$. A double sequence can be displayed as an infinite array of numbers, whose first row is the sequence $(f_{0,0}, f_{0,1}, f_{0,2}, \ldots)$, the second row is $(f_{1,0}, f_{1,1}, f_{1,2}, \ldots)$, and so on. The array can also be read by columns, and then $(f_{0,0}, f_{1,0}, f_{2,0}, \ldots)$ is the first column, $(f_{0,1}, f_{1,1}, f_{2,1}, \ldots)$ is the second column, and so on. Therefore, the index n is the *row index* and the index k is the *column index*.

We wish to describe here some sequences and double sequences of numbers, frequently occurring in the analysis of algorithms. In fact, they arise in the study of very simple and basic combinatorial problems, and therefore appear in more complex situations, so to be considered as the fundamental bricks in a solid wall.

2.1 Mappings and powers

In all branches of Mathematics two concepts appear, which have to be taken as basic: the concept of a *set* and the concept of a *mapping*. A set is a collection of objects and it must be considered a primitive concept, i.e., a concept which cannot be defined in terms of other and more elementary concepts. If $a, b, c, \ldots$ are the objects (or *elements*) in a set denoted by A, we write $A = \{a, b, c, \ldots\}$, thus giving an *extensional* definition of this particular set. If a set B is defined through a property P of its elements, we write $B = \{x \,|\, P(x) \text{ is true}\}$, thus giving an *intensional* definition of the particular set B.

If S is a finite set, then $|S|$ denotes its *cardinality* or the number of its elements: The order in which we write or consider the elements of S is irrelevant. If we wish to emphasize a particular arrangement or ordering of the elements in S, we write $(a_1, a_2, \ldots, a_n)$, if $|S| = n$ and $S = \{a_1, a_2, \ldots, a_n\}$ in any order. This is the *vector notation* and is used to represent arrangements of S. Two arrangements of S are different if and only if an index k exists, for which the elements corresponding to that index in the two arrangements are different; obviously, as sets, the two arrangements continue to be the same.

If A, B are two sets, a *mapping* or *function* from A into B, noted as $f : A \to B$, is a subset of the Cartesian product of A by B, $f \subseteq A \times B$, such that every element $a \in A$ is the first element of one and only one pair $(a, b) \in f$. The usual notation for $(a, b) \in f$ is $f(a) = b$, and b is called the *image* of a under the mapping f. The set A is the *domain* of the mapping, while B is the *range* or *codomain*. A function for which every $a_1 \neq a_2 \in A$ corresponds to pairs (a_1, b_1) and (a_2, b_2) with $b_1 \neq b_2$ is called *injective*. A function in which every $b \in B$ belongs at least to one pair (a, b) is called *surjective*. A *bijection* or *1-1 correspondence* or *1-1 mapping* is any injective function which is also surjective.

If $|A| = n$ and $|B| = m$, the cartesian product $A \times B$, i.e., the set of all the couples (a, b) with $a \in A$ and $b \in B$, contains exactly nm elements or couples. A more difficult problem is to find out how many mappings from A to B exist. We can observe that every element $a \in A$ must have its image in B; this means that we can associate to a any one of the m elements in B. Therefore, we have $m \cdot m \cdot \ldots \cdot m$ different possibilities, when the product is extended to all the n elements in A. Since all the mappings can be built in this way, we have a total of m^n different mappings from A into B. This also explains why the set of mappings from A into B is often denoted by

B^A; in fact we can write:

$$\left|B^A\right| = |B|^{|A|} = m^n.$$

This formula allows us to solve some simple combinatorial problems. For example, if we toss 5 coins, how many different configurations head/tail are possible? The five coins are the domain of our mappings and the set {head,tail} is the codomain. Therefore we have a total of $2^5 = 32$ different configurations. Similarly, if we toss three dice, the total number of configurations is $6^3 = 216$. In the same way, we can count the number of subsets in a set S having $|S| = n$. In fact, let us consider, given a subset $A \subseteq S$, the mapping $\chi_A : S \to \{0,1\}$ defined by:

$$\begin{cases} \chi_A(x) = 1 & \text{for } x \in A \\ \chi_A(x) = 0 & \text{for } x \notin A \end{cases}$$

This is called the *characteristic function* of the subset A; every two different subsets of S have different characteristic functions, and every mapping $f : S \to \{0,1\}$ is the characteristic function of some subset $A \subseteq S$, i.e., the subset $\{x \in S \,|\, f(x) = 1\}$. Therefore, there are as many subsets of S as there are characteristic functions; but these are 2^n by the formula above.

A finite set A is sometimes called an *alphabet* and its elements *symbols* or *letters*. Any sequence of letters is called a *word*; the empty sequence is the *empty word* and is denoted by ϵ. From the previous considerations, if $|A| = n$, the number of words of length m is n^m. The first part of Chapter 6 is devoted to some basic notions on special sets of words or *languages*.

2.2 Permutations

In the usual sense, a permutation of a set of objects is an arrangement of these objects in any order. For example, three objects, denoted by a, b, c, can be arranged into six different ways:

$$(a,b,c), (a,c,b), (b,a,c), (b,c,a), (c,a,b), (c,b,a).$$

A very important problem in Computer Science is "sorting": suppose we have n objects from an ordered set, usually some set of numbers or some set of strings (with the common lexicographic order); the objects are given in a random order and the problem consists in sorting them according to the given order. For example, by sorting $(60, 51, 80, 77, 44)$ we should obtain $(44, 51, 60, 77, 80)$ and the real problem is to obtain this ordering in the shortest possible time. In other words, we start with a random permutation of the n objects, and wish to arrive to their standard ordering, the one in accordance with their supposed order relation (e.g., "less than").

In order to abstract from the particular nature of the n objects, we will use the numbers $\{1, 2, \ldots, n\} = \mathbb{N}_n$, and define a *permutation* as a 1-1 mapping $\pi : \mathbb{N}_n \to \mathbb{N}_n$. By identifying a and 1, b and 2, c and 3, the six permutations of 3 objects are written:

$$\begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 3 \\ 1 & 3 & 2 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 3 \\ 2 & 1 & 3 \end{pmatrix},$$

$$\begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{pmatrix},$$

where, conventionally, the first line contains the elements in $\mathbb{N}_n$ in their proper order, and the second line contains the corresponding images. This is the usual representation for permutations, but since the first line can be understood without ambiguity, the *vector representation* for permutations is more common. This consists in writing the second line (the images) in the form of a vector. Therefore, the six permutations are $(1,2,3), (1,3,2), (2,1,3), (2,3,1), (3,1,2), (3,2,1)$, respectively.

Let us examine the permutation $\pi = (3,2,1)$ for which we have $\pi(1) = 3, \pi(2) = 2$ and $\pi(3) = 1$. If we start with the element 1 and successively apply the mapping π, we have $\pi(1) = 3, \pi(\pi(1)) = \pi(3) = 1, \pi(\pi(\pi(1))) = \pi(1) = 3$ and so on. Since the elements in $\mathbb{N}_n$ are finite, by starting with any $k \in \mathbb{N}_n$ we must obtain a finite chain of numbers, which will repeat always in the same order. These numbers are said to form a *cycle* and the permutation $(3,2,1)$ is formed by two cycles, the first one composed by 1 and 3, the second one only composed by 2. We write $(3,2,1) = (1\ 3)(2)$, where every cycle is written between parentheses and numbers are separated by blanks, to distinguish a cycle from a vector. Conventionally, a cycle is written with the smallest element first and the various cycles are arranged according to their first element. Therefore, in this *cycle representation* the six permutations are:

$$(1)(2)(3), (1)(2\ 3), (1\ 2)(3), (1\ 2\ 3), (1\ 3\ 2), (1\ 3)(2).$$

A number k for which $\pi(k) = k$ is called a *fixed point* for π. The corresponding cycles, formed by a single element, are conventionally understood, except in the identity $(1, 2, \ldots, n) = (1)(2)\cdots(n)$, in which all the elements are fixed points; the identity is simply written (1). Consequently, the usual representation of the six permutations is:

$$(1) \quad (2\ 3) \quad (1\ 2) \quad (1\ 2\ 3) \quad (1\ 3\ 2) \quad (1\ 3).$$

A permutation without any fixed point is called a *derangement*. A cycle with only two elements is called a *transposition*. The *degree* of a cycle is the number of its elements, plus one; the *degree of a permutation*

is the sum of the degrees of its cycles. The six permutations have degree 2, 3, 3, 4, 4, 3, respectively. A permutation is *even* or *odd* according to the fact that its degree is even or odd.

The permutation $(8, 9, 4, 3, 6, 1, 7, 2, 10, 5)$, in vector notation, has a cycle representation (1 8 2 9 10 5 6)(3 4), the number 7 being a fixed point. The long cycle (1 8 2 9 10 5 6) has degree 8; therefore the permutation degree is $8 + 3 + 2 = 13$ and the permutation is odd.

2.3 The group structure

Let $n \in \mathbb{N}$; P_n denotes the set of all the permutations of n elements, i.e., according to the previous sections, the set of 1-1 mappings $\pi : \mathbb{N}_n \to \mathbb{N}_n$. If $\pi, \rho \in P_n$, we can perform their *composition*, i.e., a new permutation σ defined as $\sigma(k) = \pi(\rho(k)) = (\pi \circ \rho)(k)$. An example in P_7 is:

$$\pi \circ \rho =$$
$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 1 & 5 & 6 & 7 & 4 & 2 & 3 \end{pmatrix} \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 4 & 5 & 2 & 1 & 7 & 6 & 3 \end{pmatrix}$$
$$= \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 4 & 7 & 6 & 3 & 1 & 5 & 2 \end{pmatrix}.$$

In fact, by instance, $\pi(2) = 5$ and $\rho(5) = 7$; therefore $\sigma(2) = \rho(\pi(2)) = \rho(5) = 7$, and so on. The vector representation of permutations is not particularly suited for hand evaluation of composition, although it is very convenient for computer implementation. The opposite situation occurs for cycle representation:

$$(2\ 5\ 4\ 7\ 3\ 6) \circ (1\ 4)(2\ 5\ 7\ 3) = (1\ 4\ 3\ 6\ 5)(2\ 7)$$

Cycles in the left hand member are read from left to right and by examining a cycle after the other we find the images of every element by simply remembering the cycle successor of the same element. For example, the image of 4 in the first cycle is 7; the second cycle does not contain 7, but the third cycle tells us that the image of 7 is 3. Therefore, the image of 4 is 3. Fixed points are ignored, and this is in accordance with their meaning. The composition symbol '$\circ$' is usually understood and, in reality, the simple juxtaposition of the cycles can well denote their composition.

The identity mapping acts as an identity for composition, since it is only composed by fixed points. Every permutation has an inverse, which is the permutation obtained by reading the given permutation from below, i.e., by sorting the elements in the second line, which become the elements in the first line, and then rearranging the elements in the first line into the second line. For example, the inverse of the first permutation π in the example above is:

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 1 & 6 & 7 & 5 & 2 & 3 & 4 \end{pmatrix}.$$

In fact, in cycle notation, we have:

$$(2\ 5\ 4\ 7\ 3\ 6)(2\ 6\ 3\ 7\ 4\ 5) =$$
$$= \quad (2\ 6\ 3\ 7\ 4\ 5)(2\ 5\ 4\ 7\ 3\ 6) = (1).$$

A simple observation is that the inverse of a cycle is obtained by writing its first element followed by all the other elements in reverse order. Hence, the inverse of a transposition is the same transposition.

Since composition is associative, we have proved that $(P_n, \circ)$ is a group. The group is not commutative, because, for example:

$$\rho \circ \pi = (1\ 4)(2\ 5\ 7\ 3) \circ (2\ 5\ 4\ 7\ 3\ 6)$$
$$= (1\ 7\ 6\ 2\ 4)(3\ 5) \neq \pi \circ \rho.$$

An *involution* is a permutation π such that $\pi^2 = \pi \circ \pi = (1)$. An involution can only be composed by fixed points and transpositions, because by the definition we have $\pi^{-1} = \pi$ and the above observation on the inversion of cycles shows that a cycle with more than 2 elements has an inverse which cannot coincide with the cycle itself.

Till now, we have supposed that in the cycle representation every number is only considered once. However, if we think of a permutation as the product of cycles, we can imagine that its representation is not unique and that an element $k \in \mathbb{N}_n$ can appear in more than one cycle. The representation of σ or $\pi \circ \rho$ are examples of this statement. In particular, we can obtain the *transposition representation* of a permutation; we observe that we have:

$$(2\ 6)(6\ 5)(6\ 4)(6\ 7)(6\ 3) = (2\ 5\ 4\ 7\ 3\ 6)$$

We transform the cycle into a product of transpositions by forming a transposition with the first and the last element in the cycle, and then adding other transpositions with the same first element (the last element in the cycle) and the other elements in the cycle, in the same order, as second element. Besides, we note that we can always add a couple of transpositions as (2 5)(2 5), corresponding to the two fixed points (2) and (5), and therefore adding nothing to the permutation. All these remarks show that:

- every permutation can be written as the composition of transpositions;
- this representation is not unique, but every two representations differ by an even number of transpositions;
- the minimal number of transpositions corresponding to a cycle is the degree of the cycle (except possibly for fixed points, which however always correspond to an even number of transpositions).

Therefore, we conclude that an even [odd] permutation can be expressed as the composition of an even [odd] number of transpositions. Since the composition of two even permutations is still an even permutation, the set A_n of even permutations is a subgroup of P_n and is called the *alternating subgroup*, while the whole group P_n is referred to as the *symmetric group*.

2.4 Counting permutations

How many permutations are there in P_n? If $n = 1$, we only have a single permutation (1), and if $n = 2$ we have two permutations, exactly $(1, 2)$ and $(2, 1)$. We have already seen that $|P_3| = 6$ and if $n = 0$ we consider the empty vector () as the only possible permutation, that is $|P_0| = 1$. In this way we obtain a sequence $\{1, 1, 2, 6, \ldots\}$ and we wish to obtain a formula giving us $|P_n|$, for every $n \in \mathbb{N}$.

Let $\pi \in P_n$ be a permutation and $(a_1, a_2, ..., a_n)$ be its vector representation. We can obtain a permutation in P_{n+1} by simply adding the new element $n+1$ in any position of the representation of π:

$$(n+1, a_1, a_2, \ldots, a_n) \quad (a_1, n+1, a_2, \ldots, a_n) \quad \cdots$$

$$\cdots \quad (a_1, a_2, \ldots, a_n, n+1)$$

Therefore, from any permutation in P_n we obtain $n+1$ permutations in P_{n+1}, and they are all different. Vice versa, if we start with a permutation in P_{n+1}, and eliminate the element $n + 1$, we obtain one and only one permutation in P_n. Therefore, all permutations in P_{n+1} are obtained in the way just described and are obtained only once. So we find:

$$|P_{n+1}| = (n+1)\,|P_n|$$

which is a simple recurrence relation. By unfolding this recurrence, i.e., by substituting to $|P_n|$ the same expression in $|P_{n+1}|$, and so on, we obtain:

$$\begin{aligned} |P_{n+1}| &= (n+1)|P_n| = (n+1)n|P_{n-1}| = \\ &= \cdots = \\ &= (n+1)n(n-1)\cdots 1 \times |P_0| \end{aligned}$$

Since, as we have seen, $|P_0|$=1, we have proved that the number of permutations in P_n is given by the product $n \cdot (n-1)... \cdot 2 \cdot 1$. Therefore, our sequence is:

n	0	1	2	3	4	5	6	7	8
$\|P_n\|$	1	1	2	6	24	120	720	5040	40320

As we mentioned in the Introduction, the number $n \cdot (n-1)... \cdot 2 \cdot 1$ is called *n factorial* and is denoted by $n!$. For example we have $10! = 10 \cdot 9 \cdot 8 \cdot 7 \cdot 6 \cdot 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 = 3,680,800$. Factorials grow very fast, but they are one of the most important quantities in Mathematics.

When $n \geq 2$, we can add to every permutation in P_n one transposition, say (1 2). This transforms every even permutation into an odd permutation, and vice versa. On the other hand, since $(1\ 2)^{-1} = (1\ 2)$, the transformation is its own inverse, and therefore defines a 1-1 mapping between even and odd permutations. This proves that the number of even (odd) permutations is $n!/2$.

Another simple problem is how to determine the number of involutions on n elements. As we have already seen, an involution is only composed by fixed points and transpositions (without repetitions of the elements!). If we denote by I_n the set of involutions of n elements, we can divide I_n into two subsets: I'_n is the set of involutions in which n is a fixed point, and I''_n is the set of involutions in which n belongs to a transposition, say $(k\ n)$. If we eliminate n from the involutions in I'_n, we obtain an involution of $n-1$ elements, and vice versa every involution in I'_n can be obtained by adding the fixed point n to an involution in I_{n-1}. If we eliminate the transposition $(k\ n)$ from an involution in I''_n, we obtain an involution in I_{n-2}, which contains the element $n-1$, but does not contain the element k. In all cases, however, by eliminating $(k\ n)$ from all involutions containing it, we obtain a set of involutions in a 1-1 correspondence with I_{n-2}. The element k can assume any value $1, 2, \ldots, n-1$, and therefore we obtain $(n-1)$ times $|I_{n-2}|$ involutions.

We now observe that all the involutions in I_n are obtained in this way from involutions in I_{n-1} and I_{n-2}, and therefore we have:

$$|I_n| = |I_{n-1}| + (n-1)|I_{n-2}|$$

Since $|I_0| = 1$, $|I_1| = 1$ and $|I_2| = 2$, from this recurrence relation we can successively find all the values of $|I_n|$. This sequence (see Section 4.9) is therefore:

n	0	1	2	3	4	5	6	7	8
I_n	1	1	2	4	10	26	76	232	764

We conclude this section by giving the classical computer program for generating a random permutation of the numbers $\{1, 2, \ldots, n\}$. The procedure *shuffle* receives the address of the vector, and the number of its elements; fills it with the numbers from 1 to n then uses the standard procedure *random* to produce a random permutation, which is returned in the input vector:

```
procedure shuffle( var v : vector; n : integer ) ;
var : ···;
begin
        for i := 1 to n do v[ i ] := i ;
        for i := n downto 2 do begin
            j := random( i ) + 1 ;
```

```
            a := v[ i ]; v[ i ] := v[ j ]; v[ j ] := a
        end
end {shuffle} ;
```

The procedure exchanges the last element in v with a random element in v, possibly the same last element. In this way the last element is chosen at random, and the procedure goes on with the last but one element. In this way, the elements in v are properly shuffled and eventually v contains the desired random permutation. The procedure obviously performs in linear time.

2.5 Dispositions and Combinations

Permutations are a special case of a more general situation. If we have n objects, we can wonder how many different orderings exist of k among the n objects. For example, if we have 4 objects a, b, c, d, we can make 12 different arrangements with two objects chosen from a, b, c, d. They are:

$$(a,b) \quad (a,c) \quad (a,d) \quad (b,a) \quad (b,c) \quad (b,d)$$

$$(c,a) \quad (c,b) \quad (c,d) \quad (d,a) \quad (d,b) \quad (d,c)$$

These arrangements ara called *dispositions* and, in general, we can use any one of the n objects to be first in the permutation. There remain only $n-1$ objects to be used as second element, and $n-2$ objects to be used as a third element, and so on. Therefore, the k objects can be selected in $n(n-1)...(n-k+1)$ different ways. If $D_{n,k}$ denotes the number of possible dispositions of n elements in groups of k, we have:

$$D_{n,k} = n(n-1)\cdots(n-k+1) = n^{\underline{k}}$$

The symbol $n^{\underline{k}}$ is called a *falling factorial* because it consists of k factors beginning with n and decreasing by one down to $(n-k+1)$. Obviously, $n^{\underline{n}} = n!$ and, by convention, $n^{\underline{0}} = 1$. There exists also a *rising factorial* $n^{\overline{k}} = n(n+1)\cdots(n+k-1)$, often denoted by $(n)_k$, the so-called *Pochhammer symbol*.

When in k-dispositions we do not consider the order of the elements, we obtain what are called *k-combinations*. Therefore, there are only 6 2-combinations of 4 objects, and they are:

$$\{a,b\} \quad \{a,c\} \quad \{a,d\} \quad \{b,c\} \quad \{b,d\} \quad \{c,d\}$$

Combinations are written between braces, because they are simply the subsets with k objects of a set of n objects. If $A = \{a,b,c\}$, all the possible combinations of these objects are:

$$\begin{aligned} C_{3,0} &= \emptyset \\ C_{3,1} &= \{\{a\},\{b\},\{c\}\} \\ C_{3,2} &= \{\{a,b\},\{a,c\},\{b,c\}\} \\ C_{3,3} &= \{\{a,b,c\}\} \end{aligned}$$

The number of k-combinations of n objects is denoted by $\binom{n}{k}$, which is often read "n choose k", and is called a *binomial coefficient*. By the very definition we have:

$$\binom{n}{0} = 1 \qquad \binom{n}{n} = 1 \qquad \forall n \in \mathbb{N}$$

because, given a set of n elements, the empty set is the only subset with 0 elements, and the whole set is the only subset with n elements.

The name "binomial coefficients" is due to the well-known "binomial formula":

$$(a+b)^n = \sum_{k=0}^{n} \binom{n}{k} a^k b^{n-k}$$

which is easily proved. In fact, when expanding the product $(a+b)(a+b)\cdots(a+b)$, we choose a term from each factor $(a+b)$; the resulting term $a^k b^{n-k}$ is obtained by summing over all the possible choices of k a's and $n-k$ b's, which, by the definition above, are just $\binom{n}{k}$.

There exists a simple formula to compute binomial coefficients. As we have seen, there are $n^{\underline{k}}$ different k-dispositions of n objects; by permuting the k objects in the disposition, we obtain $k!$ other dispositions with the same elements. Therefore, $k!$ dispositions correspond to a single combination and we have:

$$\binom{n}{k} = \frac{n^{\underline{k}}}{k!} = \frac{n(n-1)\ldots(n-k+1)}{k!}$$

This formula gives a simple way to compute binomial coefficients in a recursive way. In fact we have:

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

and the expression on the right is successively reduced to $\binom{r}{0}$, which is 1 as we have already seen. For example, we have:

$$\binom{7}{3} = \frac{7}{3}\binom{6}{2} = \frac{7}{3}\frac{6}{2}\binom{5}{1} = \frac{7}{3}\frac{6}{2}\frac{5}{1}\binom{4}{0} = 35$$

It is not difficult to compute a binomial coefficient such as $\binom{100}{3}$, but it is a hard job to compute $\binom{100}{97}$

by means of the above formulas. There exists, however, a symmetry formula which is very useful. Let us begin by observing that:

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

This is a very important formula by its own, and it shows that:

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

From this formula we immediately obtain $\binom{100}{97} = \binom{100}{3}$. The most difficult computing problem is the evaluation of the *central binomial coefficient* $\binom{2k}{k}$, for which symmetry gives no help.

The reader is invited to produce a computer program to evaluate binomial coefficients. He (or she) is warned not to use the formula $n!/k!(n-k)!$, which can produce very large numbers, exceeding the capacity of the computer when n, k are not small.

The definition of a binomial coefficient can be easily expanded to any real numerator:

$$\binom{r}{k} = \frac{r^{\underline{k}}}{k!} = \frac{r(r-1)\dots(r-k+1)}{k!}.$$

For example we have:

$$\binom{1/2}{3} = \frac{1/2(-1/2)(-3/2)}{3!} = \frac{1}{16}$$

but in this case the symmetry rule does not make sense. We point out that:

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

which allows us to express a binomial coefficient with negative, integer numerator as a binomial coefficient with positive numerator. This is known as *negation rule* and will be used very often.

If in a combination we are allowed to have several copies of the same element, we obtain a *combination with repetitions*. A useful exercise is to prove that the number of the k by k combinations with repetitions of n elements is:

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

2.6 The Pascal triangle

Binomial coefficients satisfy a very important recurrence relation, which we are now going to prove. As we know, $\binom{n}{k}$ is the number of the subsets with k elements of a set with n elements. We can count the number of such subsets in the following way. Let $\{a_1, a_2, \dots, a_n\}$ be the elements of the base set, and let us fix one of these elements, e.g., a_n. We can distinguish the subsets with k elements into two classes: the subsets containing a_n and the subsets that do not contain a_n. Let S^+ and S^- be these two classes. We now point out that S^+ can be seen (by eliminating a_n) as the subsets with $k-1$ elements of a set with $n-1$ elements; therefore, the number of the elements in S^+ is $\binom{n-1}{k-1}$. The class S^- can be seen as composed by the subsets with k elements of a set with $n-1$ elements, i.e., the base set minus a_n: their number is therefore $\binom{n-1}{k}$. By summing these two contributions, we obtain the recurrence relation:

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

which can be used with the initial conditions:

$$\binom{n}{0} = \binom{n}{n} = 1 \qquad \forall n \in \mathbb{N}.$$

For example, we have:

$$\begin{aligned}\binom{4}{2} &= \binom{3}{2} + \binom{3}{1} = \\ &= \binom{2}{2} + \binom{2}{1} + \binom{2}{1} + \binom{2}{0} = \\ &= 2 + 2\binom{2}{1} = 2 + 2\left(\binom{1}{1} + \binom{1}{0}\right) = \\ &= 2 + 2 \times 2 = 6.\end{aligned}$$

This recurrence is not particularly suitable for numerical computation. However, it gives a simple rule to compute successively all the binomial coefficients. Let us dispose them in an infinite array, whose rows represent the number n and whose columns represent the number k. The recurrence tells us that the element in position (n, k) is obtained by summing two elements in the previous row: the element just above the position (n, k), i.e., in position $(n-1, k)$, and the element on the left, i.e., in position $(n-1, k-1)$. The array is initially filled by 1's in the first column (corresponding to the various $\binom{n}{0}$) and the main diagonal (corresponding to the numbers $\binom{n}{n}$). See Table 2.1.

We actually obtain an infinite, lower triangular array known as the *Pascal triangle* (or Tartaglia-Pascal triangle). The symmetry rule is quite evident and a simple observation is that the sum of the elements in row n is 2^n. The proof of this fact is immediate, since

$n\backslash k$	0	1	2	3	4	5	6
0	1						
1	1	1					
2	1	2	1				
3	1	3	3	1			
4	1	4	6	4	1		
5	1	5	10	10	5	1	
6	1	6	15	20	15	6	1

Table 2.1: The Pascal triangle

a row represents the total number of the subsets in a set with n elements, and this number is obviously 2^n. The reader can try to prove that the row alternating sums, e.g., the sum $1-4+6-4+1$, equal zero, except for the first row, the row with $n = 0$.

As we have already mentioned, the numbers $c_n = \binom{2n}{n}$ are called *central binomial coefficients* and their sequence begins:

n	0	1	2	3	4	5	6	7	8
c_n	1	2	6	20	70	252	924	3432	12870

They are very important, and, for example, we can express the binomial coefficients $\binom{-1/2}{n}$ in terms of the central binomial coefficients:

$$\begin{aligned}\binom{-1/2}{n} &= \frac{(-1/2)(-3/2)\cdots(-(2n-1)/2)}{n!} = \\ &= \frac{(-1)^n 1\cdot 3\cdots(2n-1)}{2^n n!} = \\ &= \frac{(-1)^n 1\cdot 2\cdot 3\cdot 4\cdots(2n-1)\cdot(2n)}{2^n n! 2\cdot 4\cdots(2n)} = \\ &= \frac{(-1)^n (2n)!}{2^n n! 2^n(1\cdot 2\cdots n)} = \frac{(-1)^n}{4^n}\,\frac{(2n)!}{n!^2} = \\ &= \frac{(-1)^n}{4^n}\binom{2n}{n}.\end{aligned}$$

In a similar way, we can prove the following identities:

$$\begin{aligned}\binom{1/2}{n} &= \frac{(-1)^{n-1}}{4^n(2n-1)}\binom{2n}{n} \\ \binom{3/2}{n} &= \frac{(-1)^n 3}{4^n(2n-1)(2n-3)}\binom{2n}{n} \\ \binom{-3/2}{n} &= \frac{(-1)^{n-1}(2n+1)}{4^n}\binom{2n}{n}.\end{aligned}$$

An important point of this generalization is that the binomial formula can be extended to real exponents. Let us consider the function $f(x) = (1+x)^r$; it is continuous and can be differentiated as many times as we wish; in fact we have:

$$f^{(n)}(x) = \frac{d^n}{dx^n}(1+x)^r = r^{\underline{n}}(1+x)^{r-n}$$

as can be shown by mathematical induction. The first two cases are $f^{(0)} = (1+x)^r = r^{\underline{0}}(1+x)^{r-0}$, $f'(x) = r(1+x)^{r-1}$. Suppose now that the formula holds true for some $n \in \mathbb{N}$ and let us differentiate it once more:

$$\begin{aligned}f^{(n+1)}(x) &= r^{\underline{n}}(r-n)(1+x)^{r-n-1} = \\ &= r^{\underline{n+1}}(1+x)^{r-(n+1)}\end{aligned}$$

and this proves our statement. Because of that, $(1+x)^r$ has a Taylor expansion around the point $x = 0$ of the form:

$$(1+x)^r =$$

$$= f(0) + \frac{f'(0)}{1!}x + \frac{f''(0)}{2!}x^2 + \cdots + \frac{f^{(n)}(0)}{n!}x^n + \cdots.$$

The coefficient of x^n is therefore $f^{(n)}(0)/n! = r^{\underline{n}}/n! = \binom{r}{n}$, and so:

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

We conclude with the following property, which is called the *cross-product rule*:

$$\begin{aligned}\binom{n}{k}\binom{k}{r} &= \frac{n!}{k!(n-k)!}\,\frac{k!}{r!(k-r)!} = \\ &= \frac{n!}{r!(n-r)!}\,\frac{(n-r)!}{(n-k)!(k-r)!} = \\ &= \binom{n}{r}\binom{n-r}{k-r}.\end{aligned}$$

This rule, together with the symmetry and the negation rules, are the three basic properties of binomial coefficients:

$$\binom{n}{k} = \binom{n}{n-k} \qquad \binom{-n}{k} = \binom{n+k-1}{k}(-1)^k$$

$$\binom{n}{k}\binom{k}{r} = \binom{n}{r}\binom{n-r}{k-r}$$

and are to be remembered by heart.

2.7 Harmonic numbers

It is well-known that the *harmonic series*:

$$\sum_{k=1}^{\infty}\frac{1}{k} = 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \cdots$$

diverges. In fact, if we cumulate the 2^m numbers from $1/(2^m+1)$ to $1/2^{m+1}$, we obtain:

$$\frac{1}{2^m+1} + \frac{1}{2^m+2} + \cdots + \frac{1}{2^{m+1}} >$$

$$> \frac{1}{2^{m+1}} + \frac{1}{2^{m+1}} + \cdots + \frac{1}{2^{m+1}} = \frac{2^m}{2^{m+1}} = \frac{1}{2}$$

and therefore the sum cannot be limited. On the other hand we can define:

$$H_n = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n}$$

a finite, partial sum of the harmonic series. This number has a well-defined value and is called a *harmonic number*. Conventionally, we set $H_0 = 0$ and the sequence of harmonic numbers begins:

n	0	1	2	3	4	5	6	7	8
H_n	0	1	$\frac{3}{2}$	$\frac{11}{6}$	$\frac{25}{12}$	$\frac{137}{60}$	$\frac{49}{20}$	$\frac{363}{140}$	$\frac{761}{280}$

Harmonic numbers arise in the analysis of many algorithms and it is very useful to know an approximate value for them. Let us consider the series:

$$1 - \frac{1}{2} + \frac{1}{3} - \frac{1}{4} + \frac{1}{5} - \cdots = \ln 2$$

and let us define:

$$L_n = 1 - \frac{1}{2} + \frac{1}{3} - \frac{1}{4} + \cdots + \frac{(-1)^{n-1}}{n}.$$

Obviously we have:

$$H_{2n} - L_{2n} = 2\left(\frac{1}{2} + \frac{1}{4} + \cdots + \frac{1}{2n}\right) = H_n$$

or $H_{2n} = L_{2n} + H_n$, and since the series for $\ln 2$ is alternating in sign, the error committed by truncating it at any place is less than the first discarded element. Therefore:

$$\ln 2 - \frac{1}{2n} < L_{2n} < \ln 2$$

and by summing H_n to all members:

$$H_n + \ln 2 - \frac{1}{2n} < H_{2n} < H_n + \ln 2$$

Let us now consider the two cases $n = 2^{k-1}$ and $n = 2^{k-2}$:

$$H_{2^{k-1}} + \ln 2 - \frac{1}{2^k} < H_{2^k} < H_{2^{k-1}} + \ln 2$$

$$H_{2^{k-2}} + \ln 2 - \frac{1}{2^{k-1}} < H_{2^{k-1}} < H_{2^{k-2}} + \ln 2.$$

By summing and simplifying these two expressions, we obtain:

$$H_{2^{k-2}} + 2\ln 2 - \frac{1}{2^k} - \frac{1}{2^{k-1}} < H_{2^k} < H_{2^{k-2}} + 2\ln 2.$$

We can now iterate this procedure and eventually find:

$$H_{2^0} + k\ln 2 - \frac{1}{2^k} - \frac{1}{2^{k-1}} - \cdots - \frac{1}{2^1} < H_{2^k} < H_{2^0} + k\ln 2.$$

Since $H_{2^0} = H_1 = 1$, we have the limitations:

$$\ln 2^k < H_{2^k} < \ln 2^k + 1.$$

These limitations can be extended to every n, and since the values of the H_n's are increasing, this implies that a constant γ should exist ($0 < \gamma < 1$) such that:

$$H_n \to \ln n + \gamma \qquad \text{as} \qquad n \to \infty$$

This constant is called the *Euler-Mascheroni constant* and, as we have already mentioned, its value is: $\gamma \approx 0.5772156649\ldots$. Later we will prove the more accurate approximation of the H_n's we quoted in the Introduction.

The *generalized harmonic numbers* are defined as:

$$H_n^{(s)} = \frac{1}{1^s} + \frac{1}{2^s} + \frac{1}{3^s} + \cdots + \frac{1}{n^s}$$

and $H_n^{(1)} = H_n$. They are the partial sums of the series defining the Riemann ζ function:

$$\zeta(s) = \frac{1}{1^s} + \frac{1}{2^s} + \frac{1}{3^s} + \cdots$$

which can be defined in such a way that the sum actually converges except for $s = 1$ (the harmonic series). In particular we have:

$$\zeta(2) = 1 + \frac{1}{4} + \frac{1}{9} + \frac{1}{16} + \frac{1}{25} + \cdots = \frac{\pi^2}{6}$$

$$\zeta(3) = 1 + \frac{1}{8} + \frac{1}{27} + \frac{1}{64} + \frac{1}{125} + \cdots$$

$$\zeta(4) = 1 + \frac{1}{16} + \frac{1}{81} + \frac{1}{256} + \frac{1}{625} + \cdots = \frac{\pi^4}{90}$$

and in general:

$$\zeta(2n) = \frac{(2\pi)^{2n}}{2(2n)!}|B_{2n}|$$

where B_n are the Bernoulli numbers (see below). No explicit formula is known for $\zeta(2n+1)$, but numerically we have:

$$\zeta(3) \approx 1.202056903\ldots.$$

Because of the limited value of $\zeta(s)$, we can set, for large values of n:

$$H_n^{(s)} \approx \zeta(s)$$

2.8 Fibonacci numbers

At the beginning of 1200, Leonardo Fibonacci introduced in Europe the positional notation for numbers, together with the computing algorithms for performing the four basic operations. In fact, Fibonacci was

the most important mathematician in western Europe at that time. He posed the following problem: a farmer has a couple of rabbits which generates another couple after two months and, from that moment on, a new couple of rabbits every month. The new generated couples become fertile after two months, when they begin to generate a new couple of rabbits every month. The problem consists in computing how many couples of rabbits the farmer has after n months.

It is a simple matter to find the initial values: there is one couple at the beginning (1st month) and 1 in the second month. The third month the farmer has 2 couples, and 3 couples the fourth month; in fact, the first couple has generated another pair of rabbits, while the previously generated couple of rabbits has not yet become fertile. The couples become 5 on the fifth month: in fact, there are the 3 couples of the preceding month plus the newly generated couples, which are as many as there are fertile couples; but these are just the couples of two months beforehand, i.e., 2 couples. In general, at the nth month, the farmer will have the couples of the previous month plus the new couples, which are generated by the fertile couples, that is the couples he had two months before. If we denote by F_n the number of couples at the nth month, we have the *Fibonacci recurrence*:

$$F_n = F_{n-1} + F_{n-2}$$

with the initial conditions $F_1 = F_2 = 1$. By the same rule, we have $F_0 = 0$ and the sequence of *Fibonacci numbers* begins:

n	0	1	2	3	4	5	6	7	8	9	10
F_n	0	1	1	2	3	5	8	13	21	34	55

every term obtained by summing the two preceding numbers in the sequence.

Despite the small numbers appearing at the beginning of the sequence, Fibonacci numbers grow very fast, and later we will see how they grow and how they can be computed in a fast way. For the moment, we wish to show how Fibonacci numbers appear in combinatorics and in the analysis of algorithms. Suppose we have some bricks of dimensions 1×2 dm, and we wish to cover a strip 2 dm wide and n dm long by using these bricks. The problem is to know in how many different ways we can perform this covering. In Figure 2.1 we show the five ways to cover a strip 4 dm long.

If M_n is this number, we can observe that a covering of M_n can be obtained by adding vertically a brick to a covering in M_{n-1} or by adding horizontally two bricks to a covering in M_{n-2}. These are the only ways of proceeding to build our coverings, and

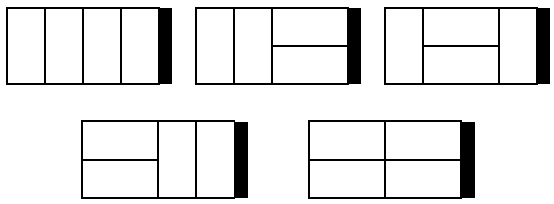


Figure 2.1: Fibonacci coverings for a strip 4 dm long

therefore we have the recurrence relation:

$$M_n = M_{n-1} + M_{n-2}$$

which is the same recurrence as for Fibonacci numbers. This time, however, we have the initial conditions $M_0 = 1$ (the empty covering is just a covering!) and $M_1 = 1$. Therefore we conclude $M_n = F_{n+1}$.

Euclid's algorithm for computing the Greatest Common Divisor (gcd) of two positive integer numbers is another instance of the appearance of Fibonacci numbers. The problem is to determine the maximal number of divisions performed by Euclid's algorithm. Obviously, this maximum is attained when every division in the process gives 1 as a quotient, since a greater quotient would drastically cut the number of necessary divisions. Let us consider two consecutive Fibonacci numbers, for example 34 and 55, and let us try to find $\gcd(34, 55)$:

$$\begin{aligned}
55 &= 1 \times 34 + 21 \\
34 &= 1 \times 21 + 13 \\
21 &= 1 \times 13 + 8 \\
13 &= 1 \times 8 + 5 \\
8 &= 1 \times 5 + 3 \\
5 &= 1 \times 3 + 2 \\
3 &= 1 \times 2 + 1 \\
2 &= 2 \times 1
\end{aligned}$$

We immediately see that the quotients are all 1 (except the last one) and the remainders are decreasing Fibonacci numbers. The process can be inverted to prove that only consecutive Fibonacci numbers enjoy this property. Therefore, we conclude that given two integer numbers, n and m, the maximal number of divisions performed by Euclid's algorithm is attained when n, m are two consecutive Fibonacci numbers and the actual number of divisions is the order of the smaller number in the Fibonacci sequence, minus 1.

2.9 Walks, trees and Catalan numbers

"Walks" or "paths" are common combinatorial objects and are defined in the following way. Let $\mathbb{Z}^2$ be

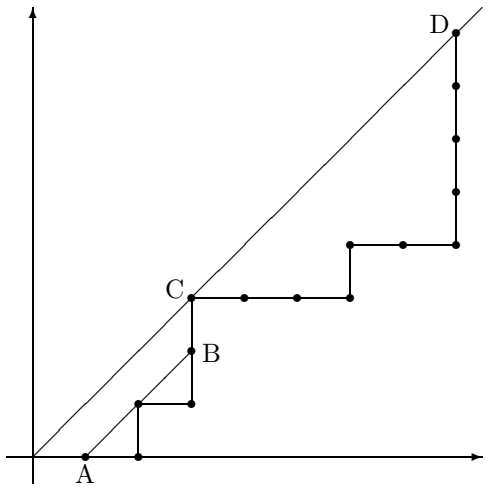


Figure 2.2: How a walk is decomposed

the *integral lattice*, i.e., the set of points in $\mathbb{R}^2$ having integer coordinates. A *walk* or *path* is a finite sequence of points in $\mathbb{Z}^2$ with the following properties:

1. the origin $(0,0)$ belongs to the walk;
2. if $(x+1, y+1)$ belongs to the walk, then either $(x, y+1)$ or $(x+1, y)$ also belongs to the walk.

A pair of points $((x,y),(x+1,y))$ is called an *east step* and a pair $((x,y),(x,y+1))$ is a *north step*. It is a simple matter to show that the number of walks composed by n steps and ending at column k (i.e., the last point is $(x,k) = (n-k,k)$) is just $\binom{n}{k}$. In fact, if we denote by $1, 2, \ldots, n$ the n steps, starting with the origin, then we can associate to any walk a subset of $\mathbb{N}_n = \{1, 2, \ldots, n\}$, that is the subset of the north-step numbers. Since the other steps should be east steps, a 1-1 correspondence exists between the walks and the subsets of $\mathbb{N}_n$ with k elements, which, as we know, are exactly $\binom{n}{k}$. This also proves, in a combinatorial way, the symmetry rule for binomial coefficients.

Among the walks, the ones that never go above the main diagonal, i.e., walks no point of which has the form (x,y) with $y > x$, are particularly important. They are called *underdiagonal walks*. Later on, we will be able to count the number of underdiagonal walks composed by n steps and ending at a horizontal distance k from the main diagonal. For the moment, we only wish to establish a recurrence relation for the number b_n of the underdiagonal walks of length $2n$ and ending on the main diagonal. In Fig. 2.2 we sketch a possible walk, where we have marked the point $C = (k,k)$ at which the walk encounters the main diagonal for the first time. We observe that the walk is decomposed into a first part (the walk between A and B) and a second part (the walk between C and D) which are underdiagonal walks of the same type. There are b_{k-1} walks of the type AB and b_{n-k} walks of the type CD and, obviously, k can be any number between 1 and n. We therefore obtain the recurrence relation:

$$b_n = \sum_{k=1}^{n} b_{k-1} b_{n-k} = \sum_{k=0}^{n-1} b_k b_{n-k-1}$$

with the initial condition $b_0 = 1$, corresponding to the empty walk or the walk composed by the only point $(0,0)$. The sequence $(b_k)_{k \in N}$ begins:

n	0	1	2	3	4	5	6	7	8
b_n	1	1	2	5	14	42	132	429	1430

and, as we shall see:

$$b_n = \frac{1}{n+1}\binom{2n}{n}.$$

The b_n's are called *Catalan numbers* and they frequently occur in the analysis of algorithms and data structures. For example, if we associate an open parenthesis to every east steps and a closed parenthesis to every north step, we obtain the number of possible parenthetizations of an expression. When we have three pairs of parentheses, the 5 possibilities are:

$$()()() \qquad ()(()) \qquad (())() \qquad (()()) \qquad ((())).$$

When we build binary trees from permutations, we do not always obtain different trees from different permutations. There are only 5 trees generated by the six permutations of $1, 2, 3$, as we show in Figure 2.3.

How many different trees exist with n nodes? If we fix our attention on the root, the left subtree has k nodes for $k = 0, 1, \ldots, n-1$, while the right subtree contains the remaining $n-k-1$ nodes. Every tree with k nodes can be combined with every tree with $n-k-1$ nodes to form a tree with n nodes, and therefore we have the recurrence relation:

$$b_n = \sum_{k=0}^{n-1} b_k b_{n-k-1}$$

which is the same recurrence as before. Since the initial condition is again $b_0 = 1$ (the empty tree) there are as many trees as there are walks.

Another kind of walks is obtained by considering steps of type $((x,y),(x+1,y+1))$, i.e., north-east steps, and of type $((x,y),(x+1,y-1))$, i.e., south-east steps. The interesting walks are those starting

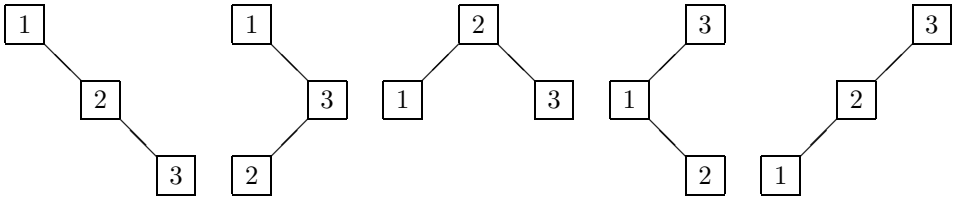


Figure 2.3: The five binary trees with three nodes

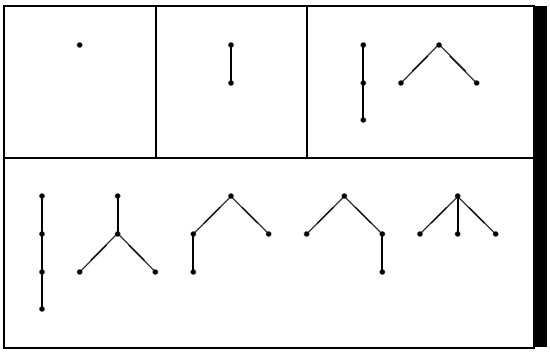


Figure 2.4: Rooted Plane Trees

from the origin and never going below the x-axis; they are called *Dyck walks*. An obvious 1-1 correspondence exists between Dyck walks and the walks considered above, and again we obtain the sequence of Catalan numbers:

n	0	1	2	3	4	5	6	7	8
b_n	1	1	2	5	14	42	132	429	1430

Finally, the concept of a *rooted planar tree* is as follows: let us consider a node, which is the *root* of the tree; if we recursively add branches to the root or to the nodes generated by previous insertions, what we obtain is a "rooted plane tree". If n denotes the number of branches in a rooted plane tree, in Fig. 2.4 we represent all the trees up to $n = 3$. Again, rooted plane trees are counted by Catalan numbers.

2.10 Stirling numbers of the first kind

About 1730, the English mathematician James Stirling was looking for a connection between powers of a number x, say x^n, and the falling factorials $x^{\underline{k}} = x(x-1)\cdots(x-k+1)$. He developed the first instances:

$$\begin{aligned} x^{\underline{1}} &= x \\ x^{\underline{2}} &= x(x-1) = x^2 - x \\ x^{\underline{3}} &= x(x-1)(x-2) = x^3 - 3x^2 + 2x \\ x^{\underline{4}} &= x(x-1)(x-2)(x-3) = \\ &= x^4 - 6x^3 + 11x^2 - 6x \end{aligned}$$

$n\backslash k$	0	1	2	3	4	5	6
0	1						
1	0	1					
2	0	1	1				
3	0	2	3	1			
4	0	6	11	6	1		
5	0	24	50	35	10	1	
6	0	120	274	225	85	15	1

Table 2.2: Stirling numbers of the first kind

and picking the coefficients in their proper order (from the smallest power to the largest) he obtained a table of integer numbers. We are mostly interested in the absolute values of these numbers, as are shown in Table 2.2. After him, these numbers are called *Stirling numbers of the first kind* and are now denoted by $\begin{bmatrix} n \\ k \end{bmatrix}$, sometimes read "$n$ cycle k", for the reason we are now going to explain.

First note that the above identities can be written:

$$x^{\underline{n}} = \sum_{k=0}^{n} \begin{bmatrix} n \\ k \end{bmatrix} (-1)^{n-k} x^k.$$

Let us now observe that $x^{\underline{n}} = x^{\underline{n-1}}(x-n+1)$ and therefore we have:

$$\begin{aligned} x^{\underline{n}} &= (x-n+1)\sum_{k=0}^{n-1} \begin{bmatrix} n-1 \\ k \end{bmatrix} (-1)^{n-k-1} x^k = \\ &= \sum_{k=0}^{n-1} \begin{bmatrix} n-1 \\ k \end{bmatrix} (-1)^{n-k-1} x^{k+1} - \\ &\quad - \sum_{k=0}^{n-1} (n-1) \begin{bmatrix} n-1 \\ k \end{bmatrix} (-1)^{n-k-1} x^k = \\ &= \sum_{k=0}^{n} \begin{bmatrix} n-1 \\ k-1 \end{bmatrix} (-1)^{n-k} x^k + \end{aligned}$$

$$+\sum_{k=0}^{n}(n-1)\begin{bmatrix} n-1 \\ k \end{bmatrix}(-1)^{n-k}x^k.$$

We performed the change of variable $k \to k-1$ in the first sum and then extended both sums from 0 to n. This identity is valid for every value of x, and therefore we can equate its coefficients to those of the previous, general Stirling identity, thus obtaining the recurrence relation:

$$\begin{bmatrix} n \\ k \end{bmatrix} = (n-1)\begin{bmatrix} n-1 \\ k \end{bmatrix} + \begin{bmatrix} n-1 \\ k-1 \end{bmatrix}.$$

This recurrence, together with the initial conditions:

$$\begin{bmatrix} n \\ n \end{bmatrix} = 1, \forall n \in \mathbb{N} \qquad \text{and} \qquad \begin{bmatrix} n \\ 0 \end{bmatrix} = 0, \forall n > 0,$$

completely defines the Stirling number of the first kind.

What is a possible combinatorial interpretation of these numbers? Let us consider the permutations of n elements and count the permutations having exactly k cycles, whose set will be denoted by $S_{n,k}$. If we fix any element, say the last element n, we observe that the permutations in $S_{n,k}$ can have n as a fixed point, or not. When n is a fixed point and we eliminate it, we obtain a permutation with $n-1$ elements having exactly $k-1$ cycles; vice versa, any such permutation gives a permutation in $S_{n,k}$ with n as a fixed point if we add (n) to it. Therefore, there are $|S_{n-1,k-1}|$ such permutations in $S_{n,k}$. When n is not a fixed point and we eliminate it from the permutation, we obtain a permutation with $n-1$ elements and k cycles. However, the same permutation is obtained several times, exactly $n-1$ times, since n can occur after any other element in the standard cycle representation (it can never occur as the first element in a cycle, by our conventions). For example, all the following permutations in $S_{5,2}$ produce the same permutation in $S_{4,2}$:

$$(1\,2\,3)(4\,5) \quad (1\,2\,3\,5)(4) \quad (1\,2\,5\,3)(4) \quad (1\,5\,2\,3)(4).$$

The process can be inverted and therefore we have:

$$|S_{n,k}| = (n-1)|S_{n-1,k}| + |S_{n-1,k-1}|$$

which is just the recurrence relation for the Stirling numbers of the first kind. If we now prove that also the initial conditions are the same, we conclude that $|S_{n,k}| = \begin{bmatrix} n \\ k \end{bmatrix}$. First we observe that $S_{n,n}$ is only composed by the identity, the only permutation having n cycles, i.e., n fixed points; so $|S_{n,n}| = 1, \forall n \in \mathbb{N}$. Moreover, for $n \geq 1$, $S_{n,0}$ is empty, because every permutation contains at least one cycle, and so $|S_{n,0}| = 0$. This concludes the proof.

As an immediate consequence of this reasoning, we find:

$$\sum_{k=0}^{n}\begin{bmatrix} n \\ k \end{bmatrix} = n!,$$

i.e., the row sums of the Stirling triangle of the first kind equal $n!$, because they correspond to the total number of permutations of n objects. We also observe that:

- $\begin{bmatrix} n \\ 1 \end{bmatrix} = (n-1)!$; in fact, $S_{n,1}$ is composed by all the permutations having a single cycle; this begins by 1 and is followed by any permutations of the $n-1$ remaining numbers;
- $\begin{bmatrix} n \\ n-1 \end{bmatrix} = \binom{n}{2}$; in fact, $S_{n,n-1}$ contains permutations having all fixed points except a single transposition; but this transposition can only be formed by taking two elements among $1, 2, \ldots, n$, which is done in $\binom{n}{2}$ different ways;
- $\begin{bmatrix} n \\ 2 \end{bmatrix} = (n-1)!H_{n-1}$; returning to the numerical definition, the coefficient of x^2 is a sum of products, in each of which a positive integer is missing.

2.11 Stirling numbers of the second kind

James Stirling also tried to invert the process described in the previous section, that is he was also interested in expressing ordinary powers in terms of falling factorials. The first instances are:

$$\begin{aligned} x^1 &= x^{\underline{1}} \\ x^2 &= x^{\underline{1}} + x^{\underline{2}} = x + x(x-1) \\ x^3 &= x^{\underline{1}} + 3x^{\underline{2}} + x^{\underline{3}} = x + 3x(x-1) + \\ &\quad + x(x-1)(x-2) \\ x^4 &= x^{\underline{1}} + 6x^{\underline{2}} + 7x^{\underline{3}} + x^{\underline{4}} \end{aligned}$$

The coefficients can be arranged into a triangular array, as shown in Table 2.3, and are called *Stirling numbers of the second kind*. The usual notation for them is $\left\{ {n \atop k} \right\}$, often read "$n$ subset k", for the reason we are going to explain.

Stirling's identities can be globally written:

$$x^n = \sum_{k=0}^{n}\left\{ {n \atop k} \right\} x^{\underline{k}}.$$

We obtain a recurrence relation in the following way:

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

$n\backslash k$	0	1	2	3	4	5	6
0	1						
1	0	1					
2	0	1	1				
3	0	1	3	1			
4	0	1	7	6	1		
5	0	1	15	25	10	1	
6	0	1	31	90	65	15	1

Table 2.3: Stirling numbers of the second kind

$$= \sum_{k=0}^{n-1} \left\{ {n-1 \atop k} \right\} x^{\underline{k+1}} + \sum_{k=0}^{n-1} k \left\{ {n-1 \atop k} \right\} x^{\underline{k}} =$$
$$= \sum_{k=0}^{n} \left\{ {n-1 \atop k-1} \right\} x^{\underline{k}} + \sum_{k=0}^{n} k \left\{ {n-1 \atop k} \right\} x^{\underline{k}}$$

where, as usual, we performed the change of variable $k \to k-1$ and extended the two sums from 0 to n. The identity is valid for every $x \in \mathbb{R}$, and therefore we can equate the coefficients of $x^{\underline{k}}$ in this and the above identity, thus obtaining the recurrence relation:

$$\left\{ {n \atop k} \right\} = k \left\{ {n-1 \atop k} \right\} + \left\{ {n-1 \atop k-1} \right\}$$

which is slightly different from the recurrence for the Stirling numbers of the first kind. Here we have the initial conditions:

$$\left\{ {n \atop n} \right\} = 1, \forall n \in \mathbb{N} \qquad \text{and} \qquad \left\{ {n \atop 0} \right\} = 0, \forall n \geq 1.$$

These relations completely define the Stirling triangle of the second kind. Every row of the triangle determines a polynomial; for example, from row 4 we obtain: $S_4(w) = w + 7w^2 + 6w^3 + w^4$ and it is called the 4-th *Stirling polynomial*.

Let us now look for a combinatorial interpretation of these numbers. If $\mathbb{N}_n$ is the usual set $\{1, 2, \ldots, n\}$, we can study the partitions of $\mathbb{N}_n$ into k disjoint, non-empty subsets. For example, when $n = 4$ and $k = 2$, we have the following 7 partitions:

$$\{1\} \cup \{2,3,4\} \quad \{1,2\} \cup \{3,4\} \quad \{1,3\} \cup \{2,4\}$$
$$\{1,4\} \cup \{2,3\} \quad \{1,2,3\} \cup \{4\}$$
$$\{1,2,4\} \cup \{3\} \quad \{1,3,4\} \cup \{2\}.$$

If $P_{n,k}$ is the corresponding set, we now count $|P_{n,k}|$ by fixing an element in $\mathbb{N}_n$, say the last element n. The partitions in $P_{n,k}$ can contain n as a singleton (i.e., as a subset with n as its only element) or can contain n as an element in a larger subset. In the former case, by eliminating $\{n\}$ we obtain a partition in $P_{n-1,k-1}$ and, obviously, all partitions in $P_{n-1,k-1}$ can be obtained in such a way. When n belongs to a larger set, we can eliminate it obtaining a partition in $P_{n-1,k}$; however, the same partition is obtained several times, exactly by eliminating n from any of the k subsets containing it in the various partitions. For example, the following three partitions in $P_{5,3}$ all produce the same partition in $P_{4,3}$:

$$\{1,2,5\} \cup \{3\} \cup \{4\} \quad \{1,2\} \cup \{3,5\} \cup \{4\}$$
$$\{1,2\} \cup \{3\} \cup \{4,5\}$$

This proves the recurrence relation:

$$|P_{n,k}| = k|P_{n-1,k}| + |P_{n-1,k-1}|$$

which is the same recurrence as for the Stirling numbers of the second kind. As far as the initial conditions are concerned, we observe that there is only one partition of $\mathbb{N}_n$ composed by n subsets, i.e., the partition containing n singletons; therefore $|P_{n,n}| = 1, \forall n \in \mathbb{N}$ (in the case $n = 0$ the empty set is the only partition of the empty set). When $n \geq 1$, there is no partition of $\mathbb{N}_n$ composed by 0 subsets, and therefore $|P_{n,0}| = 0$. We can conclude that $|P_{n,k}|$ coincides with the corresponding Stirling number of the second kind, and use this fact for observing that:

- $\left\{ {n \atop 1} \right\} = 1, \forall n \geq 1$. In fact, the only partition of $\mathbb{N}_n$ in a single subset is when the subset coincides with $\mathbb{N}_n$;
- $\left\{ {n \atop 2} \right\} = 2^{n-1}, \forall n \geq 2$. When the partition is only composed by two subsets, the first one uniquely determines the second. Let us suppose that the first subset always contains 1. By eliminating 1, we obtain as first subset all the subsets in $\mathbb{N}_n \setminus \{1\}$, except this last whole set, which would correspond to an empty second set. This proves the identity;
- $\left\{ {n \atop n-1} \right\} = \binom{n}{2}, \forall n \in \mathbb{N}$. In any partition with one subset with 2 elements, and all the others singletons, the two elements can be chosen in $\binom{n}{2}$ different ways.

2.12 Bell and Bernoulli numbers

If we sum the rows of the Stirling triangle of the second kind, we find a sequence:

n	0	1	2	3	4	5	6	7	8
$\mathcal{B}_n$	1	1	2	5	15	52	203	877	4140

which represents the total number of partitions relative to the set $\mathbb{N}_n$. For example, the five partitions of a set with three elements are:

$$\{1,2,3\} \quad \{1\} \cup \{2,3\} \quad \{1,2\} \cup \{3\}$$

$$\{1,3\}\cup\{2\} \quad \{1\}\cup\{2\}\cup\{3\}\,.$$

The numbers in this sequence are called *Bell numbers* and are denoted by $\mathcal{B}_n$; by definition we have:

$$\mathcal{B}_n = \sum_{k=0}^{n} \left\{ {n \atop k} \right\}.$$

Bell numbers grow very fast; however, since $\left\{ {n \atop k} \right\} \leq \left[{n \atop k} \right]$, for every value of n and k (a subset in $P_{n,k}$ corresponds to one or more cycles in $S_{n,k}$), we always have $\mathcal{B}_n \leq n!$, and in fact $\mathcal{B}_n < n!$ for every $n > 1$.

Another frequently occurring sequence is obtained by ordering the subsets appearing in the partitions of $\mathbb{N}_n$. For example, the partition $\{1\}\cup\{2\}\cup\{3\}$ can be ordered in $3! = 6$ different ways, and $\{1,2\}\cup\{3\}$ can be ordered in $2! = 2$ ways, i.e., $\{1,2\}\cup\{3\}$ and $\{3\}\cup\{1,2\}$. These are called *ordered partitions*, and their number is denoted by $\mathcal{O}_n$. By the previous example, we easily see that $\mathcal{O}_3 = 13$, and the sequence begins:

n	0	1	2	3	4	5	6	7
$\mathcal{O}_n$	1	1	3	13	75	541	4683	47293

Because of this definition, the numbers $\mathcal{O}_n$ are called *ordered Bell numbers* and we have:

$$\mathcal{O}_n = \sum_{k=0}^{n} \left\{ {n \atop k} \right\} k!;$$

this shows that $\mathcal{O}_n \geq n!$, and, in fact, $\mathcal{O}_n > n!, \forall n > 1$.

Another combinatorial interpretation of the ordered Bell numbers is as follows. Let us fix an integer $n \in \mathbb{N}$ and for every $k \leq n$ let A_k be any multiset with n elements containing at least once all the numbers $1, 2, \ldots, k$. The number of all the possible orderings of the A_k's is just the nth ordered Bell number. For example, when $n = 3$, the possible multisets are: $\{1,1,1\}, \{1,1,2\}, \{1,2,2\}, \{1,2,3\}$. Their possible orderings are given by the following 7 vectors:

$$(1,1,1) \quad (1,1,2) \quad (1,2,1) \quad (2,1,1)$$

$$(1,2,2) \quad (2,1,2) \quad (2,2,1)$$

plus the six permutations of the set $\{1,2,3\}$. These orderings are called *preferential arrangements*.

We can find a 1-1 correspondence between the orderings of set partitions and preferential arrangements. If $(a_1, a_2, \ldots, a_n)$ is a preferential arrangement, we build the corresponding ordered partition by setting the element 1 in the a_1th subset, 2 in the a_2th subset, and so on. If k is the largest number in the arrangement, we exactly build k subsets. For example, the partition corresponding to $(1,2,2,1)$ is $\{1,4\}\cup\{2,3\}$, while the partition corresponding to $(2,1,1,2)$ is $\{2,3\}\cup\{1,4\}$, whose ordering is different. This construction can be easily inverted and since it is injective, we have proved that it is actually a 1-1 correspondence. Because of that, ordered Bell numbers are also called *preferential arrangement numbers*.

We conclude this section by introducing another important sequence of numbers. These are (positive or negative) rational numbers and therefore they cannot correspond to any counting problem, i.e., their combinatorial interpretation cannot be direct. However, they arise in many combinatorial problems and therefore they should be examined here, for the moment only introducing their definition. The *Bernoulli numbers* are implicitly defined by the recurrence relation:

$$\sum_{k=0}^{n} \binom{n+1}{k} B_k = \delta_{n,0}.$$

No initial condition is necessary, because for $n = 0$ we have $\binom{1}{0} B_0 = 1$, i.e., $B_0 = 1$. This is the starting value, and B_1 is obtained by setting $n = 1$ in the recurrence relation:

$$\binom{2}{0} B_0 + \binom{2}{1} B_1 = 0.$$

We obtain $B_1 = -1/2$, and we now have a formula for B_2:

$$\binom{3}{0} B_0 + \binom{3}{1} B_1 + \binom{3}{2} B_2 = 0.$$

By performing the necessary computations, we find $B_2 = 1/6$, and we can go on successively obtaining all the possible values for the B_n's. The first twelve values are as follows:

n	0	1	2	3	4	5	6
B_n	1	$-1/2$	$1/6$	0	$-1/30$	0	$1/42$

n	7	8	9	10	11	12
B_n	0	$-1/30$	0	$5/66$	0	$-691/2730$

Except for B_1, all the other values of B_n for odd n are zero. Initially, Bernoulli numbers seem to be small, but as n grows, they become extremely large in modulus, but, apart from the zero values, they are alternatively one positive and one negative. These and other properties of the Bernoulli numbers are not easily proven in a direct way, i.e., from their definition. However, we'll see later how we can arrange things in such a way that everything becomes accessible to us.

Chapter 3

Formal power series

3.1 Definitions for formal power series

Let $\mathbb{R}$ be the field of real numbers and let t be any indeterminate over $\mathbb{R}$, i.e., a symbol different from any element in $\mathbb{R}$. A *formal power series* (f.p.s.) over $\mathbb{R}$ in the indeterminate t is an expression:

$$f(t) = f_0 + f_1 t + f_2 t^2 + f_3 t^3 + \cdots + f_n t^n + \cdots = \sum_{k=0}^{\infty} f_k t^k$$

where $f_0, f_1, f_2, \ldots$ are all real numbers. The same definition applies to every set of numbers, in particular to the field of rational numbers $\mathbb{Q}$ and to the field of complex numbers $\mathbb{C}$. The developments we are now going to see, and depending on the field structure of the numeric set, can be easily extended to every field $\mathbf{F}$ of 0 characteristic. The set of formal power series over $\mathbf{F}$ in the indeterminate t is denoted by $\mathbf{F}[\![t]\!]$. The use of a particular indeterminate t is irrelevant, and there exists an obvious 1-1 correspondence between, say, $\mathbf{F}[\![t]\!]$ and $\mathbf{F}[\![y]\!]$; it is simple to prove that this correspondence is indeed an isomorphism. In order to stress that our results are substantially independent of the particular field $\mathbf{F}$ and of the particular indeterminate t, we denote $\mathbf{F}[\![t]\!]$ by $\mathcal{F}$, but the reader can think of $\mathcal{F}$ as of $\mathbb{R}[\![t]\!]$. In fact, in combinatorial analysis and in the analysis of algorithms the coefficients $f_0, f_1, f_2, \ldots$ of a formal power series are mostly used to count objects, and therefore they are positive integer numbers, or, in some cases, positive rational numbers (e.g., when they are the coefficients of an exponential generating function. See below and Section 4.1).

If $f(t) \in \mathcal{F}$, the *order* of $f(t)$, denoted by $\mathrm{ord}(f(t))$, is the smallest index r for which $f_r \neq 0$. The set of all f.p.s. of order exactly r is denoted by $\mathcal{F}_r$ or by $\mathbf{F}_r[\![t]\!]$. The formal power series $0 = 0 + 0t + 0t^2 + 0t^3 + \cdots$ has infinite order.

If $(f_0, f_1, f_2, \ldots) = (f_k)_{k \in N}$ is a sequence of (real) numbers, there is no substantial difference between the sequence and the f.p.s. $\sum_{k=0}^{\infty} f_k t^k$, which will be called the *(ordinary) generating function* of the sequence. The term *ordinary* is used to distinguish these functions from *exponential generating functions*, which will be introduced in the next chapter. The indeterminate t is used as a "place-marker", i.e., a symbol to denote the place of the element in the sequence. For example, in the f.p.s. $1 + t + t^2 + t^3 + \cdots$, corresponding to the sequence $(1, 1, 1, \ldots)$, the term $t^5 = 1 \cdot t^5$ simply denotes that the element in position 5 (starting from 0) in the sequence is the number 1. Although our study of f.p.s. is mainly justified by the development of a generating function theory, we dedicate the present chapter to the general theory of f.p.s., and postpone the study of generating functions to the next chapter.

There are two main reasons why f.p.s. are more easily studied than sequences:

1. the algebraic structure of f.p.s. is very well understood and can be developed in a standard way;

2. many f.p.s. can be "abbreviated" by expressions easily manipulated by elementary algebra.

The present chapter is devoted to these algebraic aspects of f.p.s.. For example, we will prove that the series $1 + t + t^2 + t^3 + \cdots$ can be conveniently abbreviated as $1/(1-t)$, and from this fact we will be able to infer that the series has a f.p.s. inverse, which is $1 - t + 0t^2 + 0t^3 + \cdots$.

We conclude this section by defining the concept of a *formal Laurent (power) series* (f.L.s.), as an expression:

$$\begin{aligned} g(t) &= g_{-m} t^{-m} + g_{-m+1} t^{-m+1} + \cdots + g_{-1} t^{-1} + \\ &\quad + g_0 + g_1 t + g_2 t^2 + \cdots = \sum_{k=-m}^{\infty} g_k t^k. \end{aligned}$$

The set of f.L.s. strictly contains the set of f.p.s.. For a f.L.s. $g(t)$ the order can be negative; when the order of $g(t)$ is non-negative, then $g(t)$ is actually a f.p.s.. We observe explicitly that an expression as $\sum_{k=-\infty}^{\infty} f_k t^k$ does *not* represent a f.L.s..

3.2 The basic algebraic structure

The set $\mathcal{F}$ of f.p.s. can be embedded into several algebraic structures. We are now going to define the most common one, which is related to the usual concept of sum and (Cauchy) product of series. Given two f.p.s. $f(t) = \sum_{k=0}^{\infty} f_k t^k$ and $g(t) = \sum_{k=0}^{\infty} g_k t^k$, the *sum* of $f(t)$ and $g(t)$ is defined as:

$$f(t) + g(t) = \sum_{k=0}^{\infty} f_k t^k + \sum_{k=0}^{\infty} g_k t^k = \sum_{k=0}^{\infty} (f_k + g_k) t^k.$$

From this definition, it immediately follows that $\mathcal{F}$ is a commutative group with respect to the sum. The associative and commutative laws directly follow from the analogous properties in the field $\mathbf{F}$; the identity is the f.p.s. $0 = 0 + 0t + 0t^2 + 0t^3 + \cdots$, and the opposite series of $f(t) = \sum_{k=0}^{\infty} f_k t^k$ is the series $-f(t) = \sum_{k=0}^{\infty} (-f_k) t^k$.

Let us now define the *Cauchy product* of $f(t)$ by $g(t)$:

$$f(t)g(t) = \left(\sum_{k=0}^{\infty} f_k t^k \right) \left(\sum_{k=0}^{\infty} g_k t^k \right) =$$

$$= \sum_{k=0}^{\infty} \left(\sum_{j=0}^{k} f_j g_{k-j} \right) t^k$$

Because of the form of the t^k coefficient, this is also called the *convolution* of $f(t)$ and $g(t)$. It is a good idea to write down explicitly the first terms of the Cauchy product:

$$\begin{aligned} f(t)g(t) \quad = \quad & f_0 g_0 + (f_0 g_1 + f_1 g_0) t + \\ & + (f_0 g_2 + f_1 g_1 + f_2 g_0) t^2 + \\ & + (f_0 g_3 + f_1 g_2 + f_2 g_1 + f_3 g_0) t^3 + \cdots \end{aligned}$$

This clearly shows that the product is commutative and it is a simple matter to prove that the identity is the f.p.s. $1 = 1 + 0t + 0t^2 + 0t^3 + \cdots$. The distributive law is a consequence of the distributive law valid in $\mathbf{F}$. In fact, we have:

$$\begin{aligned} & (f(t) + g(t))h(t) = \\ = \quad & \sum_{k=0}^{\infty} \left(\sum_{j=0}^{k} (f_j + g_j) h_{k-j} \right) t^k = \\ = \quad & \sum_{k=0}^{\infty} \left(\sum_{j=0}^{k} f_j h_{k-j} \right) t^k + \sum_{k=0}^{\infty} \left(\sum_{j=0}^{k} g_j h_{k-j} \right) t^k = \\ = \quad & f(t)h(t) + g(t)h(t) \end{aligned}$$

Finally, we can prove that $\mathcal{F}$ does not contain any zero divisor. If $f(t)$ and $g(t)$ are two f.p.s. different from zero, then we can suppose that $\mathrm{ord}(f(t)) = k_1$ and $\mathrm{ord}(g(t)) = k_2$, with $0 \leq k_1, k_2 < \infty$. This means $f_{k_1} \neq 0$ and $g_{k_2} \neq 0$; therefore, the product $f(t)g(t)$ has the term of degree $k_1 + k_2$ with coefficient $f_{k_1} g_{k_2} \neq 0$, and so it cannot be zero. We conclude that $(\mathcal{F}, +, \cdot)$ is an integrity domain.

The previous reasoning also shows that, in general, we have:

$$\mathrm{ord}(f(t)g(t)) = \mathrm{ord}(f(t)) + \mathrm{ord}(g(t))$$

The order of the identity 1 is obviously 0; if $f(t)$ is an invertible element in $\mathcal{F}$, we should have $f(t)f(t)^{-1} = 1$ and therefore $\mathrm{ord}(f(t)) = 0$. On the other hand, if $f(t) \in \mathcal{F}_0$, i.e., $f(t) = f_0 + f_1 t + f_2 t^2 + f_3 t^3 + \cdots$ with $f_0 \neq 0$, we can easily prove that $f(t)$ is invertible. In fact, let $g(t) = f(t)^{-1}$ so that $f(t)g(t) = 1$. From the explicit expression for the Cauchy product, we can determine the coefficients of $g(t)$ by solving the infinite system of linear equations:

$$\begin{cases} f_0 g_0 = 1 \\ f_0 g_1 + f_1 g_0 = 0 \\ f_0 g_2 + f_1 g_1 + f_2 g_0 = 0 \\ \cdots \end{cases}$$

The system can be solved in a simple way, starting with the first equation and going on one equation after the other. Explicitly, we obtain:

$$g_0 = f_0^{-1} \quad g_1 = -\frac{f_1}{f_0^2} \quad g_2 = -\frac{f_1^2}{f_0^3} - \frac{f_2}{f_0^2} \quad \cdots$$

and therefore $g(t) = f(t)^{-1}$ is well defined. We conclude stating the result just obtained: a f.p.s. is invertible if and only if its order is 0. Because of that, $\mathcal{F}_0$ is also called the *set of invertible f.p.s.*. According to standard terminology, the elements of $\mathcal{F}_0$ are called the *units* of the integrity domain.

As a simple example, let us compute the inverse of the f.p.s. $1 - t = 1 - t + 0t^2 + 0t^3 + \cdots$. Here we have $f_0 = 1, f_1 = -1$ and $f_k = 0, \forall k > 1$. The system becomes:

$$\begin{cases} g_0 = 1 \\ g_1 - g_0 = 0 \\ g_2 - g_1 = 0 \\ \cdots \end{cases}$$

and we easily obtain that all the g_j's $(j = 0, 1, 2, \ldots)$ are 1. Therefore the inverse f.p.s. we are looking for is $1 + t + t^2 + t^3 + \cdots$. The usual notation for this fact is:

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

It is well-known that this identity is only valid for $-1 < t < 1$, when t is a variable and f.p.s. are interpreted as functions. In our formal approach, however, these considerations are irrelevant and the identity is valid from a purely formal point of view.

3.3 Formal Laurent Series

In the first section of this Chapter, we introduced the concept of a *formal Laurent series*, as an extension of the concept of a f.p.s.; if $a(t) = \sum_{k=m}^{\infty} a_k t^k$ and $b(t) = \sum_{k=n}^{\infty} b_k t^k$ $(m, n \in \mathbb{Z})$, are two f.L.s., we can define the sum and the Cauchy product:

$$\begin{aligned} a(t) + b(t) &= \sum_{k=m}^{\infty} a_k t^k + \sum_{k=n}^{\infty} b_k t^k = \\ &= \sum_{k=p}^{\infty} (a_k + b_k) t^k \\ a(t)b(t) &= \left(\sum_{k=m}^{\infty} a_k t^k \right) \left(\sum_{k=n}^{\infty} b_k t^k \right) = \\ &= \sum_{k=q}^{\infty} \left(\sum_{i+j=k} a_i b_j \right) t^k \end{aligned}$$

where $p = \min(m, n)$ and $q = m + n$. As we did for f.p.s., it is not difficult to find out that these operations enjoy the usual properties of sum and product, and if we denote by $\mathcal{L}$ the set of f.L.s., we have that $(\mathcal{L}, +, \cdot)$ is a field. The only point we should formally prove is that every f.L.s. $a(t) = \sum_{k=m}^{\infty} a_k t^k \neq 0$ has an inverse f.L.s. $b(t) = \sum_{k=-m}^{\infty} b_k t^k$. However, this is proved in the same way we proved that every f.p.s. in $\mathcal{F}_0$ has an inverse. In fact we should have:

$$\begin{aligned} &a_m b_{-m} = 1 \\ &a_m b_{-m+1} + a_{m+1} b_{-m} = 0 \\ &a_m b_{-m+2} + a_{m+1} b_{-m+1} + a_{m+2} b_{-m} = 0 \\ &\cdots \end{aligned}$$

By solving the first equation, we find $b_{-m} = a_m^{-1}$; then the system can be solved one equation after the other, by substituting the values obtained up to the moment. Since $a_m b_{-m}$ is the coefficient of t^0, we have $a(t)b(t) = 1$ and the proof is complete.

We can now show that $(\mathcal{L}, +, \cdot)$ is the smallest field containing the integrity domain $(\mathcal{F}, +, \cdot)$, thus characterizing the set of f.L.s. in an algebraic way. From Algebra we know that given an integrity domain $(K, +, \cdot)$ the smallest field $(F, +, \cdot)$ containing $(K, +, \cdot)$ can be built in the following way: let us define an equivalence relation $\sim$ on the set $K \times K$:

$$(a, b) \sim (c, d) \iff ad = bc;$$

if we now set $F = K \times K/\sim$, the set F with the operations $+$ and $\cdot$ defined as the extension of $+$ and $\cdot$ in K is the field we are searching for. This is just the way in which the field $\mathbb{Q}$ of rational numbers is constructed from the integrity domain $\mathbb{Z}$ of integers numbers, and the field of rational functions is built from the integrity domain of the polynomials.

Our aim is to show that the field $(\mathcal{L}, +, \cdot)$ of f.L.s. is isomorphic with the field constructed in the described way starting with the integrity domain of f.p.s.. Let $\widehat{\mathcal{L}} = \mathcal{F} \times \mathcal{F}$ be the set of pairs of f.p.s.; we begin by showing that for every $(f(t), g(t)) \in \widehat{\mathcal{L}}$ there exists a pair $(a(t), b(t)) \in \widehat{\mathcal{L}}$ such that $(f(t), g(t)) \sim (a(t), b(t))$ (i.e., $f(t)b(t) = g(t)a(t)$) and at least one between $a(t)$ and $b(t)$ belongs to $\mathcal{F}_0$. In fact, let $p = \min(\operatorname{ord}(f(t)), \operatorname{ord}(g(t)))$ and let us define $a(t), b(t)$ as $f(t) = t^p a(t)$ and $g(t) = t^p b(t)$; obviously, either $a(t) \in \mathcal{F}_0$ or $b(t) \in \mathcal{F}_0$ or both are invertible f.p.s.. We now have:

$$b(t)f(t) = b(t)t^p a(t) = t^p b(t)a(t) = g(t)a(t)$$

and this shows that $(a(t), b(t)) \sim (f(t), g(t))$.

If $b(t) \in \mathcal{F}_0$, then $a(t)/b(t) \in \mathcal{F}$ and is uniquely determined by $a(t), b(t)$; in this case, therefore, our assert is proved. So, let us now suppose that $b(t) \notin \mathcal{F}_0$; then we can write $b(t) = t^m v(t)$, where $v(t) \in \mathcal{F}_0$. We have $a(t)/v(t) = \sum_{k=0}^{\infty} d_k t^k \in \mathcal{F}_0$, and consequently let us consider the f.L.s. $l(t) = \sum_{k=0}^{\infty} d_k t^{k-m}$; by construction, it is uniquely determined by $a(t), b(t)$ or also by $f(t), g(t)$. It is now easy to see that $l(t)$ is the inverse of the f.p.s. $b(t)/a(t)$ in the sense of f.L.s. as considered above, and our proof is complete. This shows that the correspondence is a 1-1 correspondence between $\mathcal{L}$ and $\widehat{\mathcal{L}}$ preserving the inverse, so it is now obvious that the correspondence is also an isomorphism between $(\mathcal{L}, +, \cdot)$ and $(\widehat{\mathcal{L}}, +, \cdot)$.

Because of this result, we can identify $\widehat{\mathcal{L}}$ and $\mathcal{L}$ and assert that $(\mathcal{L}, +, \cdot)$ is indeed the smallest field containing $(\mathcal{F}, +, \cdot)$. From now on, the set $\widehat{\mathcal{L}}$ will be ignored and we will always refer to $\mathcal{L}$ as the field of f.L.s..

3.4 Operations on formal power series

Besides the four basic operations: addition, subtraction, multiplication and division, it is possible to consider other operations on $\mathcal{F}$, only a few of which can be extended to $\mathcal{L}$.

The most important operation is surely taking a *power* of a f.p.s.; if $p \in \mathbb{N}$ we can recursively define:

$$\begin{cases} f(t)^0 = 1 & \text{if } p = 0 \\ f(t)^p = f(t)f(t)^{p-1} & \text{if } p > 0 \end{cases}$$

and observe that $\operatorname{ord}(f(t)^p) = p \operatorname{ord}(f(t))$. Therefore, $f(t)^p \in \mathcal{F}_0$ if and only if $f(t) \in \mathcal{F}_0$; on the other hand, if $f(t) \notin \mathcal{F}_0$, then the order of $f(t)^p$ becomes larger and larger and goes to ∞ when $p \to \infty$. This property will be important in our future developments, when we will reduce many operations to

infinite sums involving the powers $f(t)^p$ with $p \in \mathbb{N}$. If $f(t) \notin \mathcal{F}_0$, i.e., $\mathrm{ord}(f(t)) > 0$, these sums involve elements of larger and larger order, and therefore for every index k we can determine the coefficient of t^k by only a finite number of terms. This assures that our definitions will be good definitions.

We wish also to observe that taking a positive integer power can be easily extended to $\mathcal{L}$; in this case, when $\mathrm{ord}(f(t)) < 0$, $\mathrm{ord}(f(t)^p)$ decreases, but remains always finite. In particular, for $g(t) = f(t)^{-1}$, $g(t)^p = f(t)^{-p}$, and powers can be extended to all integers $p \in \mathbb{Z}$.

When the exponent p is a real or complex number whatsoever, we should restrict $f(t)^p$ to the case $f(t) \in \mathcal{F}_0$; in fact, if $f(t) = t^m g(t)$, we would have: $f(t)^p = (t^m g(t))^p = t^{mp} g(t)^p$; however, t^{mp} is an expression without any mathematical sense. Instead, if $f(t) \in \mathcal{F}_0$, let us write $f(t) = f_0 + \widehat{v}(t)$, with $\mathrm{ord}(\widehat{v}(t)) > 0$. For $v(t) = \widehat{v}(t)/f_0$, we have by Newton's rule:

$$f(t)^p = (f_0 + \widehat{v}(t))^p = f_0^p (1+v(t))^p = f_0^p \sum_{k=0}^{\infty} \binom{p}{k} v(t)^k,$$

which can be assumed as a definition. In the last expression, we can observe that: i) $f_0^p \in \mathbb{C}$; ii) $\binom{p}{k}$ is defined for every value of p, k being a non-negative integer; iii) $v(t)^k$ is well-defined by the considerations above and $\mathrm{ord}(v(t)^k)$ grows indefinitely, so that for every k the coefficient of t^k is obtained by a finite sum. We can conclude that $f(t)^p$ is well-defined.

Particular cases are $p = -1$ and $p = 1/2$. In the former case, $f(t)^{-1}$ is the inverse of the f.p.s. $f(t)$. We have already seen a method for computing $f(t)^{-1}$, but now we obtain the following formula:

$$f(t)^{-1} = \frac{1}{f_0} \sum_{k=0}^{\infty} \binom{-1}{k} v(t)^k = \frac{1}{f_0} \sum_{k=0}^{\infty} (-1)^k v(t)^k.$$

For $p = 1/2$, we obtain a formula for the square root of a f.p.s.:

$$\begin{aligned} f(t)^{1/2} &= \sqrt{f(t)} = \sqrt{f_0} \sum_{k=0}^{\infty} \binom{1/2}{k} v(t)^k = \\ &= \sqrt{f_0} \sum_{k=0}^{\infty} \frac{(-1)^{k-1}}{4^k (2k-1)} \binom{2k}{k} v(t)^k. \end{aligned}$$

In Section 3.12, we will see how $f(t)^p$ can be obtained computationally without actually performing the powers $v(t)^k$. We conclude by observing that this more general operation of taking the power $p \in \mathbb{R}$ cannot be extended to f.L.s.: in fact, we would have smaller and smaller terms t^k ($k \to -\infty$) and therefore the resulting expression cannot be considered an actual f.L.s., which requires a term with smallest degree.

By applying well-known rules of the exponential and logarithmic functions, we can easily define the corresponding operations for f.p.s., which however, as will be apparent, cannot be extended to f.L.s.. For the *exponentiation* we have for $f(t) \in \mathcal{F}_0, f(t) = f_0 + v(t)$:

$$e^{f(t)} = \exp(f_0 + v(t)) = e^{f_0} \sum_{k=0}^{\infty} \frac{v(t)^k}{k!}.$$

Again, since $v(t) \notin \mathcal{F}_0$, the order of $v(t)^k$ increases with k and the sums necessary to compute the coefficient of t^k are always finite. The formula makes clear that exponentiation can be performed on every $f(t) \in \mathcal{F}$, and when $f(t) \notin \mathcal{F}_0$ the factor e^{f_0} is not present.

For the *logarithm*, let us suppose $f(t) \in \mathcal{F}_0, f(t) = f_0 + \widehat{v}(t), v(t) = \widehat{v}(t)/f_0$; then we have:

$$\begin{aligned} \ln(f_0 + \widehat{v}(t)) &= \ln f_0 + \ln(1 + v(t)) = \\ &= \ln f_0 + \sum_{k=1}^{\infty} (-1)^{k+1} \frac{v(t)^k}{k}. \end{aligned}$$

In this case, for $f(t) \notin \mathcal{F}_0$, we cannot define the logarithm, and this shows an asymmetry between exponential and logarithm.

Another important operation is *differentiation*:

$$Df(t) = \frac{d}{dt} f(t) = \sum_{k=1}^{\infty} k f_k t^{k-1} = f'(t).$$

This operation can be performed on every $f(t) \in \mathcal{L}$, and a very important observation is the following:

Theorem 3.4.1 *For every $f(t) \in \mathcal{L}$, its derivative $f'(t)$ does not contain any term in t^{-1}.*

Proof: In fact, by the general rule, the term in t^{-1} should have been originated by the constant term (i.e., the term in t^0) in $f(t)$, but the product by k reduces it to 0. ∎

This fact will be the basis for very important results on the theory of f.p.s. and f.L.s. (see Section 3.8).

Another operation is *integration*; because indefinite integration leaves a constant term undefined, we prefer to introduce and use only *definite integration*; for $f(t) \in \mathcal{F}$ this is defined as:

$$\int_0^t f(\tau) d\tau = \sum_{k=0}^{\infty} f_k \int_0^t \tau^k d\tau = \sum_{k=0}^{\infty} \frac{f_k}{k+1} t^{k+1}.$$

Our purely formal approach allows us to exchange the integration and summation signs; in general, as we know, this is only possible when the convergence is uniform. By this definition, $\int_0^t f(\tau) d\tau$ never belongs to $\mathcal{F}_0$. Integration can be extended to f.L.s. with an

obvious exception: because integration is the inverse operation of differentiation, we cannot apply integration to a f.L.s. containing a term in t^{-1}. Formally, from the definition above, such a term would imply a division by 0, and this is not allowed. In all the other cases, integration does not create any problem.

3.5 Composition

A last operation on f.p.s. is so important that we dedicate to it a complete section. The operation is the *composition* of two f.p.s.. Let $f(t) \in \mathcal{F}$ and $g(t) \notin \mathcal{F}_0$, then we define the "composition" of $f(t)$ by $g(t)$ as the f.p.s:

$$f(g(t)) = \sum_{k=0}^{\infty} f_k g(t)^k.$$

This definition justifies the fact that $g(t)$ cannot belong to F_0; in fact, otherwise, infinite sums were involved in the computation of $f(g(t))$. In connection with the composition of f.p.s., we will use the following notation:

$$f(g(t)) = \left[f(y) \;\middle|\; y = g(t) \right]$$

which is intended to mean the result of substituting $g(t)$ to the indeterminate y in the f.p.s. $f(y)$. The indeterminate y is a dummy symbol; it should be different from t in order not to create any ambiguity, but it can be substituted by any other symbol. Because every $f(t) \notin \mathcal{F}_0$ is characterized by the fact that $g(0) = 0$, we will always understand that, in the notation above, the f.p.s. $g(t)$ is such that $g(0) = 0$.

The definition can be extended to every $f(t) \in \mathcal{L}$, but the function $g(t)$ has always to be such that $\mathrm{ord}(g(t)) > 0$, otherwise the definition would imply infinite sums, which we avoid because, by our formal approach, we do not consider any convergence criterion.

The f.p.s. in $\mathcal{F}_1$ have a particular relevance for composition. They are called *quasi-units* or *delta series*. First of all, we wish to observe that the f.p.s. $t \in \mathcal{F}_1$ acts as an identity for composition. In fact $\left[y \;\middle|\; y = g(t) \right] = g(t)$ and $\left[f(y) \;\middle|\; y = t \right] = f(t)$, and therefore t is a left and right identity. As a second fact, we show that a f.p.s. $f(t)$ has an inverse with respect to composition if and only if $f(t) \in \mathcal{F}_1$. Note that $g(t)$ is the inverse of $f(t)$ if and only if $f(g(t)) = t$ and $g(f(t)) = t$. From this, we deduce immediately that $f(t) \notin \mathcal{F}_0$ and $g(t) \notin \mathcal{F}_0$. On the other hand, it is clear that $\mathrm{ord}(f(g(t))) = \mathrm{ord}(f(t))\mathrm{ord}(g(t))$ by our initial definition, and since $\mathrm{ord}(t) = 1$ and $\mathrm{ord}(f(t)) > 0$, $\mathrm{ord}(g(t)) > 0$, we must have $\mathrm{ord}(f(t)) = \mathrm{ord}(g(t)) = 1$.

Let us now come to the main part of the proof and consider the set $\mathcal{F}_1$ with the operation of composition $\circ$; composition is always associative and therefore $(\mathcal{F}_1, \circ)$ is a group if we prove that every $f(t) \in \mathcal{F}_1$ has a left (or right) inverse, because the theory assures that the other inverse exists and coincides with the previously found inverse. Let $f(t) = f_1 t + f_2 t^2 + f_3 t^3 + \cdots$ and $g(t) = g_1 t + g_2 t^2 + g_3 t^3 + \cdots$; we have:

$$\begin{aligned} f(g(t)) &= f_1(g_1 t + g_2 t^2 + g_3 t^3 + \cdots) + \\ &\quad + f_2(g_1^2 t^2 + 2 g_1 g_2 t^3 + \cdots) + \\ &\quad + f_3(g_1^3 t^3 + \cdots) + \cdots = \\ &= f_1 g_1 t + (f_1 g_2 + f_2 g_1^2) t^2 + \\ &\quad + (f_1 g_3 + 2 f_2 g_1 g_2 + f_3 g_1^3) t^3 + \cdots \\ &= t \end{aligned}$$

In order to determine $g(t)$ we have to solve the system:

$$\begin{cases} f_1 g_1 = 1 \\ f_1 g_2 + f_2 g_1^2 = 0 \\ f_1 g_3 + 2 f_2 g_1 g_2 + f_3 g_1^3 = 0 \\ \cdots \end{cases}$$

The first equation gives $g_1 = 1/f_1$; we can substitute this value in the second equation and obtain a value for g_2; the two values for g_1 and g_2 can be substituted in the third equation and obtain a value for g_3. Continuing in this way, we obtain the value of all the coefficients of $g(t)$, and therefore $g(t)$ is determined in a unique way. In fact, we observe that, by construction, in the kth equation, g_k appears in linear form and its coefficient is always f_1. Being $f_1 \neq 0$, g_k is unique even if the other g_r $(r < k)$ appear with powers.

The f.p.s. $g(t)$ such that $f(g(t)) = t$, and therefore such that $g(f(t)) = t$ as well, is called the *compositional inverse* of $f(t)$. In the literature, it is usually denoted by $\overline{f}(t)$ or $f^{[-1]}(t)$; we will adopt the first notation. Obviously, $\overline{\overline{f}}(t) = f(t)$, and sometimes $\overline{f}(t)$ is also called the *reverse* of $f(t)$. Given $f(t) \in \mathcal{F}_1$, the determination of its compositional inverse is one of the most interesting problems in the theory of f.p.s. or f.L.s.; it was solved by Lagrange and we will discuss it in the following sections. Note that, in principle, the g_k's can be computed by solving the system above; this, however, is too complicated and nobody will follow that way, unless for exercising.

3.6 Coefficient extraction

If $f(t) \in \mathcal{L}$, or in particular $f(t) \in \mathcal{F}$, the notation $[t^n]f(t)$ indicates the *extraction of the coefficient of* t^n from $f(t)$, and therefore we have $[t^n]f(t) = f_n$. In this sense, $[t^n]$ can be seen as a mapping: $[t^n] : \mathcal{L} \to \mathbb{R}$ or $[t^n] : \mathcal{L} \to \mathbb{C}$, according to what is the field underlying the set $\mathcal{L}$ or $\mathcal{F}$. Because of that, $[t^n]$

(linearity)	$[t^n](\alpha f(t)+\beta g(t))$	$=$	$\alpha[t^n]f(t)+\beta[t^n]g(t)$	$(K1)$
(shifting)	$[t^n]tf(t)$	$=$	$[t^{n-1}]f(t)$	$(K2)$
(differentiation)	$[t^n]f'(t)$	$=$	$(n+1)[t^{n+1}]f(t)$	$(K3)$
(convolution)	$[t^n]f(t)g(t)$	$=$	$\sum_{k=0}^{n}[t^k]f(t)[t^{n-k}]g(t)$	$(K4)$
(composition)	$[t^n]f(g(t))$	$=$	$\sum_{k=0}^{\infty}([y^k]f(y))[t^n]g(t)^k$	$(K5)$

Table 3.1: The rules for coefficient extraction

is called an *operator* and exactly the *"coefficient of" operator* or, more simply, the *coefficient operator*.

In Table 3.1 we state formally the main properties of this operator, by collecting what we said in the previous sections. We observe that $\alpha,\beta\in\mathbb{R}$ or $\alpha,\beta\in\mathbb{C}$ are any constants; the use of the indeterminate y is only necessary not to confuse the action on different f.p.s.; because $g(0)=0$ in composition, the last sum is actually finite. Some points require more lengthy comments. The property of shifting can be easily generalized to $[t^n]t^kf(t)=[t^{n-k}]f(t)$ and also to negative powers: $[t^n]f(t)/t^k=[t^{n+k}]f(t)$. These rules are very important and are often applied in the theory of f.p.s. and f.L.s.. In the former case, some care should be exercised to see whether the properties remain in the realm of $\mathcal{F}$ or go beyond it, invading the domain of $\mathcal{L}$, which can be not always correct. The property of differentiation for $n=1$ gives $[t^{-1}]f'(t)=0$, a situation we already noticed. The operator $[t^{-1}]$ is also called the *residue* and is noted as "res"; so, for example, people write $\operatorname{res}f'(t)=0$ and some authors use the notation $\operatorname{res}t^{n+1}f(t)$ for $[t^n]f(t)$.

We will have many occasions to apply rules $(K1)\div(K5)$ of coefficient extraction. However, just to give a meaningful example, let us find the coefficient of t^n in the series expansion of $(1+\alpha t)^r$, when α and r are two real numbers whatsoever. Rule $(K3)$ can be written in the form $[t^n]f(t)=\frac{1}{n}[t^{n-1}]f'(t)$ and we can successively apply this form to our case:

$$\begin{aligned}[t^n](1+\alpha t)^r &= \frac{r\alpha}{n}[t^{n-1}](1+\alpha t)^{r-1}=\\ &= \frac{r\alpha}{n}\frac{(r-1)\alpha}{n-1}[t^{n-2}](1+\alpha t)^{r-2}=\cdots=\\ &= \frac{r\alpha}{n}\frac{(r-1)\alpha}{n-1}\cdots\frac{(r-n+1)\alpha}{1}[t^0](1+\alpha t)^{r-n}=\\ &= \alpha^n\binom{r}{n}[t^0](1+\alpha t)^{r-n}.\end{aligned}$$

We now observe that $[t^0](1+\alpha t)^{r-n}=1$ because of our observations on f.p.s. operations. Therefore, we conclude with the so-called *Newton's rule*:

$$[t^n](1+\alpha t)^r=\binom{r}{n}\alpha^n$$

which is one of the most frequently used results in coefficient extraction. Let us remark explicitly that when $r=-1$ (the *geometric series*) we have:

$$\begin{aligned}[t^n]\frac{1}{1+\alpha t} &= \binom{-1}{n}\alpha^n=\\ &= \binom{1+n-1}{n}(-1)^n\alpha^n=(-\alpha)^n.\end{aligned}$$

A simple, but important use of Newton's rule concerns the extraction of the coefficient of t^n from the inverse of a trinomial at^2+bt+c, in the case it is reducible, i.e., it can be written $(1+\alpha t)(1+\beta t)$; obviously, we can always reduce the constant c to 1; by the linearity rule, it can be taken outside the "coefficient of" operator. Therefore, our aim is to compute:

$$[t^n]\frac{1}{(1+\alpha t)(1+\beta t)}$$

with $\alpha\neq\beta$, otherwise Newton's rule would be immediately applicable. The problem can be solved by using the technique of *partial fraction expansion*. We look for two constants A and B such that:

$$\begin{aligned}\frac{1}{(1+\alpha t)(1+\beta t)} &= \frac{A}{1+\alpha t}+\frac{B}{1+\beta t}=\\ &= \frac{A+A\beta t+B+B\alpha t}{(1+\alpha t)(1+\beta t)};\end{aligned}$$

if two such constants exist, the numerator in the first expression should equal the numerator in the last one, independently of t, or, if one so prefers, for every value of t. Therefore, the term $A+B$ should be equal to 1, while the term $(A\beta+B\alpha)t$ should always be 0. The values for A and B are therefore the solution of the linear system:

$$\begin{cases}A+B=1\\ A\beta+B\alpha=0\end{cases}$$

The discriminant of this system is $\alpha - \beta$, which is always different from 0, because of our hypothesis $\alpha \neq \beta$. The system has therefore only one solution, which is $A = \alpha/(\alpha-\beta)$ and $B = -\beta/(\alpha-\beta)$. We can now substitute these values in the expression above:

$$\begin{aligned}
&[t^n]\frac{1}{(1+\alpha t)(1+\beta t)} = \\
&= [t^n]\frac{1}{\alpha-\beta}\left(\frac{\alpha}{1+\alpha t} - \frac{\beta}{1+\beta t}\right) = \\
&= \frac{1}{\alpha-\beta}\left([t^n]\frac{\alpha}{1+\alpha t} - [t^n]\frac{\beta}{1+\beta t}\right) = \\
&= \frac{\alpha^{n+1}-\beta^{n+1}}{\alpha-\beta}(-1)^n
\end{aligned}$$

Let us now consider a trinomial $1 + bt + ct^2$ for which $\Delta = b^2 - 4c < 0$ and $b \neq 0$. The trinomial is irreducible, but we can write:

$$[t^n]\frac{1}{1+bt+ct^2} =$$

$$= [t^n]\frac{1}{\left(1 - \frac{-b+i\sqrt{|\Delta|}}{2}t\right)\left(1 - \frac{-b-i\sqrt{|\Delta|}}{2}t\right)}$$

This time, a partial fraction expansion does not give a simple closed form for the coefficients, however, we can apply the formula above in the form:

$$[t^n]\frac{1}{(1-\alpha t)(1-\beta t)} = \frac{\alpha^{n+1}-\beta^{n+1}}{\alpha-\beta}.$$

Since α and β are complex numbers, the resulting expression is not very appealing. We can try to give it a better form. Let us set $\alpha = \left(-b + i\sqrt{|\Delta|}\right)/2$, so α is always contained in the positive imaginary halfplane. This implies $0 < \arg(\alpha) < \pi$ and we have:

$$\begin{aligned}
\alpha - \beta &= -\frac{b}{2} + i\frac{\sqrt{|\Delta|}}{2} + \frac{b}{2} + i\frac{\sqrt{|\Delta|}}{2} = \\
&= i\sqrt{|\Delta|} = i\sqrt{4c-b^2}
\end{aligned}$$

If $\theta = \arg(\alpha)$ and:

$$\rho = |\alpha| = \sqrt{\frac{b^2}{4} - \frac{4c-b^2}{4}} = \sqrt{c}$$

we can set $\alpha = \rho e^{i\theta}$ and $\beta = \rho e^{-i\theta}$. Consequently:

$$\alpha^{n+1} - \beta^{n+1} = \rho^{n+1}\left(e^{i(n+1)\theta} - e^{-i(n+1)\theta}\right) =$$

$$= 2i\rho^{n+1}\sin(n+1)\theta$$

and therefore:

$$[t^n]\frac{1}{1+bt+ct^2} = \frac{2(\sqrt{c})^{n+1}\sin(n+1)\theta}{\sqrt{4c-b^2}}$$

At this point we only have to find the value of θ. Obviously:

$$\theta = \arctan\left(\frac{\sqrt{|\Delta|}}{2}\Big/\frac{-b}{2}\right) + k\pi = \arctan\frac{\sqrt{4c-b^2}}{-b} + k\pi$$

When $b < 0$, we have $0 < \arctan\left(-\sqrt{4c-b^2}\right)/2 < \pi/2$, and this is the correct value for θ. However, when $b > 0$, the principal branch of arctan is negative, and we should set $\theta = \pi + \arctan\left(-\sqrt{4c-b^2}\right)/2$. As a consequence, we have:

$$\theta = \arctan\frac{\sqrt{4c-b^2}}{-b} + C$$

where $C = \pi$ if $b > 0$ and $C = 0$ if $b < 0$.

An interesting and non-trivial example is given by:

$$\begin{aligned}
\sigma_n &= [t^n]\frac{1}{1-3t+3t^2} = \\
&= \frac{2(\sqrt{3})^{n+1}\sin((n+1)\arctan(\sqrt{3}/3))}{\sqrt{3}} = \\
&= 2(\sqrt{3})^n \sin\frac{(n+1)\pi}{6}
\end{aligned}$$

These coefficients have the following values:

$n = 12k$	$\sigma_n = \sqrt{3}^{12k} = 729^k$
$n = 12k+1$	$\sigma_n = \sqrt{3}^{12k+2} = 3 \times 729^k$
$n = 12k+2$	$\sigma_n = 2\sqrt{3}^{12k+2} = 6 \times 729^k$
$n = 12k+3$	$\sigma_n = \sqrt{3}^{12k+4} = 9 \times 729^k$
$n = 12k+4$	$\sigma_n = \sqrt{3}^{12k+4} = 9 \times 729^k$
$n = 12k+5$	$\sigma_n = 0$
$n = 12k+6$	$\sigma_n = -\sqrt{3}^{12k+6} = -27 \times 729^k$
$n = 12k+7$	$\sigma_n = -\sqrt{3}^{12k+8} = -81 \times 729^k$
$n = 12k+8$	$\sigma_n = -2\sqrt{3}^{12k+8} = -162 \times 729^k$
$n = 12k+9$	$\sigma_n = -\sqrt{3}^{12k+10} = -243 \times 729^k$
$n = 12k+10$	$\sigma_n = -\sqrt{3}^{12k+10} = -243 \times 729^k$
$n = 12k+11$	$\sigma_n = 0$

3.7 Matrix representation

Let $f(t) \in \mathcal{F}_0$; with the coefficients of $f(t)$ we form the following infinite lower triangular matrix (or array) $D = (d_{n,k})_{n,k\in N}$: column 0 contains the coefficients $f_0, f_1, f_2, \ldots$ in this order; column 1 contains the same coefficients shifted down by one position and $d_{0,1} = 0$; in general, column k contains the coefficients of $f(t)$ shifted down k positions, so that the first k positions are 0. This definition can be summarized in the formula $d_{n,k} = f_{n-k}, \forall n, k \in \mathbb{N}$. For a reason which will be apparent only later, the array

D will be denoted by $(f(t),1)$:

$$D=(f(t),1)=\begin{pmatrix} f_0 & 0 & 0 & 0 & 0 & \cdots \\ f_1 & f_0 & 0 & 0 & 0 & \cdots \\ f_2 & f_1 & f_0 & 0 & 0 & \cdots \\ f_3 & f_2 & f_1 & f_0 & 0 & \cdots \\ f_4 & f_3 & f_2 & f_1 & f_0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}.$$

If $(f(t),1)$ and $(g(t),1)$ are the matrices corresponding to the two f.p.s. $f(t)$ and $g(t)$, we are interested in finding out what is the matrix obtained by multiplying the two matrices with the usual row-by-column product. This product will be denoted by $(f(t),1)\cdot(g(t),1)$, and it is immediate to see what its generic element $d_{n,k}$ is. The row n in $(f(t),1)$ is, by definition, $\{f_n,f_{n-1},f_{n-2},\ldots\}$, and column k in $(g(t),1)$ is $\{0,0,\ldots,0,g_1,g_2,\ldots\}$ where the number of leading 0's is just k. Therefore we have:

$$d_{n,k}=\sum_{j=0}^{\infty}f_{n-j}g_{j-k}$$

if we conventionally set $g_r=0,\forall r<0$, When $k=0$, we have $d_{n,0}=\sum_{j=0}^{\infty}f_{n-j}g_j=\sum_{j=0}^{n}f_{n-j}g_j$, and therefore column 0 contains the coefficients of the convolution $f(t)g(t)$. When $k=1$ we have $d_{n,1}=\sum_{j=0}^{\infty}f_{n-j}g_{j-1}=\sum_{j=0}^{n-1}f_{n-1-j}g_j$, and this is the coefficient of t^{n-1} in the convolution $f(t)g(t)$. Proceeding in the same way, we see that column k contains the coefficients of the convolution $f(t)g(t)$ shifted down k positions. Therefore we conclude:

$$(f(t),1)\cdot(g(t),1)=(f(t)g(t),1)$$

and this shows that there exists a group isomorphism between $(\mathcal{F}_0,\cdot)$ and the set of matrices $(f(t),1)$ with the row-by-column product. In particular, $(1,1)$ is the identity (in fact, it corresponds to the identity matrix) and $(f(t)^{-1},1)$ is the inverse of $(f(t),1)$.

Let us now consider a f.p.s. $f(t)\in\mathcal{F}_1$ and let us build an infinite lower triangular matrix in the following way: column k contains the coefficients of $f(t)^k$ in their proper order:

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & \cdots \\ 0 & f_1 & 0 & 0 & 0 & \cdots \\ 0 & f_2 & f_1^2 & 0 & 0 & \cdots \\ 0 & f_3 & 2f_1f_2 & f_1^3 & 0 & \cdots \\ 0 & f_4 & 2f_1f_3+f_2^2 & 3f_1^2f_2 & f_1^4 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}.$$

The matrix will be denoted by $(1,f(t)/t)$ and we are interested to see how the matrix $(1,g(t)/t)\cdot(1,f(t)/t)$ is composed when $f(t),g(t)\in\mathcal{F}_1$.

If $\left(\widehat{f}_{n,k}\right)_{n,k\in N}=(1,f(t)/t)$, by definition we have:

$$\widehat{f}_{n,k}=[t^n]f(t)^k$$

and therefore the generic element $d_{n,k}$ of the product is:

$$\begin{aligned} d_{n,k} &= \sum_{j=0}^{\infty}\widehat{g}_{n,j}\widehat{f}_{j,k}=\sum_{j=0}^{\infty}[t^n]g(t)^j[y^j]f(y)^k= \\ &= [t^n]\sum_{j=0}^{\infty}\left([y^j]f(t)^k\right)g(t)^j=[t^n]f(g(t))^k. \end{aligned}$$

In other words, column k in $(1,g(t)/t)\cdot(1,f(t)/t)$ is the kth power of the composition $f(g(t))$, and we can conclude:

$$(1,g(t)/t)\cdot(1,f(t)/t)=(1,f(g(t))/t).$$

Clearly, the identity $t\in\mathcal{F}_1$ corresponds to the matrix $(1,t/t)=(1,1)$, the identity matrix, and this is sufficient to prove that the correspondence $f(t)\leftrightarrow(1,f(t)/t)$ is a group isomorphism.

Row-by-column product is surely the basic operation on matrices and its extension to infinite, lower triangular arrays is straight-forward, because the sums involved in the product are actually finite. We have shown that we can associate every f.p.s. $f(t)\in\mathcal{F}_0$ to a particular matrix $(f(t),1)$ (let us denote by A the set of such arrays) in such a way that $(\mathcal{F}_0,\cdot)$ is isomorphic to $(A,\cdot)$, and the Cauchy product becomes the row-by-column product. Besides, we can associate every f.p.s. $g(t)\in\mathcal{F}_1$ to a matrix $(1,g(t)/t)$ (let us call B the set of such matrices) in such a way that $(\mathcal{F}_1,\circ)$ is isomorphic to $(B,\cdot)$, and the composition of f.p.s. becomes again the row-by-column product. This reveals a connection between the Cauchy product and the composition: in the Chapter on Riordan Arrays we will explore more deeply this connection; for the moment, we wish to see how this observation yields to a computational method for evaluating the compositional inverse of a f.p.s. in $\mathcal{F}_1$.

3.8 Lagrange inversion theorem

Given an infinite, lower triangular array of the form $(1,f(t)/t)$, with $f(t)\in\mathcal{F}_1$, the inverse matrix $(1,g(t)/t)$ is such that $(1,g(t)/t)\cdot(1,f(t)/t)=(1,1)$, and since the product results in $(1,f(g(t))/t)$ we have $f(g(t))=t$. In other words, because of the isomorphism we have seen, the inverse matrix for $(1,f(t)/t)$ is just the matrix corresponding to the compositional inverse of $f(t)$. As we have already said, Lagrange

found a noteworthy formula for the coefficients of this compositional inverse. We follow the more recent proof of Stanley, which points out the purely formal aspects of Lagrange's formula. Indeed, we will prove something more, by finding the exact form of the matrix $(1, g(t)/t)$, inverse of $(1, f(t)/t)$. As a matter of fact, we state what the form of $(1, g(t)/t)$ should be and then verify that it is actually so.

Let $D = (d_{n,k})_{n,k\in N}$ be defined as:

$$d_{n,k} = \frac{k}{n}[t^{n-k}]\left(\frac{t}{f(t)}\right)^n.$$

Because $f(t)/t \in \mathcal{F}_0$, the power $(t/f(t))^k = (f(t)/t)^{-k}$ is well-defined; in order to show that $(d_{n,k})_{n,k\in N} = (1, g(t)/t)$ we only have to prove that $D \cdot (1, f(t)/t) = (1, 1)$, because we already know that the compositional inverse of $f(t)$ is unique. The generic element $v_{n,k}$ of the row-by-column product $D \cdot (1, f(t)/t)$ is:

$$\begin{aligned} v_{n,k} &= \sum_{j=0}^{\infty} d_{n,j}[y^j]f(y)^k = \\ &= \sum_{j=0}^{\infty} \frac{j}{n}[t^{n-j}]\left(\frac{t}{f(t)}\right)^n [y^j]f(y)^k. \end{aligned}$$

By the rule of differentiation for the coefficient of operator, we have:

$$j[y^j]f(y)^k = [y^{j-1}]\frac{d}{dy}f(y)^k = k[y^j]yf'(y)f(y)^{k-1}.$$

Therefore, for $v_{n,k}$ we have:

$$\begin{aligned} v_{n,k} &= \frac{k}{n}\sum_{j=0}^{\infty}[t^{n-j}]\left(\frac{t}{f(t)}\right)^n [y^j]yf'(y)f(y)^{k-1} = \\ &= \frac{k}{n}[t^n]\left(\frac{t}{f(t)}\right)^n tf'(t)f(t)^{k-1}. \end{aligned}$$

In fact, the factor k/n does not depend on j and can be taken out of the summation sign; the sum is actually finite and is the term of the convolution appearing in the last formula. Let us now distinguish between the case $k = n$ and $k \neq n$. When $k = n$ we have:

$$\begin{aligned} v_{n,n} &= [t^n]t^n tf(t)^{-1}f'(t) = \\ &= [t^0]f'(t)\left(\frac{t}{f(t)}\right) = 1; \end{aligned}$$

in fact, $f'(t) = f_1 + 2f_2t + 3f_3t^2 + \cdots$ and, being $f(t)/t \in \mathcal{F}_0$, $(f(t)/t)^{-1} = (f_1 + f_2t + f_3t^2 + \cdots)^{-1} = f_1^{-1} + \cdots$; therefore, the constant term in $f'(t)(t/f(t))$ is $f_1/f_1 = 1$. When $k \neq n$:

$$\begin{aligned} v_{n,k} &= \frac{k}{n}[t^n]t^n tf(t)^{k-n-1}f'(t) = \\ &= \frac{k}{n}[t^{-1}]\frac{1}{k-n}\frac{d}{dt}\left(f(t)^{k-n}\right) = 0; \end{aligned}$$

in fact, $f(t)^{k-n}$ is a f.L.s. and, as we observed, the residue of its derivative should be zero. This proves that $D \cdot (1, f(t)/t) = (1, 1)$ and therefore D is the inverse of $(1, f(t)/t)$.

If $\overline{f}(t)$ is the compositional inverse of $f(t)$, the column 1 gives us the value of its coefficients; by the formula for $d_{n,k}$ we have:

$$\overline{f}_n = [t^n]\overline{f}(t) = d_{n,1} = \frac{1}{n}[t^{n-1}]\left(\frac{t}{f(t)}\right)^n$$

and this is the celebrated *Lagrange Inversion Formula* (LIF). The other columns give us the coefficients of the powers $\overline{f}(t)^k$, for which we have:

$$[t^n]\overline{f}(t)^k = \frac{k}{n}[t^{n-k}]\left(\frac{t}{f(t)}\right)^n.$$

Many times, there is another way for applying the LIF. Suppose we have a functional equation $w = t\phi(w)$, where $\phi(t) \in \mathcal{F}_0$, and we wish to find the f.p.s. $w = w(t)$ satisfying this functional equation. Clearly $w(t) \in \mathcal{F}_1$ and if we set $f(y) = y/\phi(y)$, we also have $f(t) \in \mathcal{F}_1$. However, the functional equation can be written $f(w(t)) = t$, and this shows that $w(t)$ is the compositional inverse of $f(t)$. We therefore know that $w(t)$ is uniquely determined and the LIF gives us:

$$[t^n]w(t) = \frac{1}{n}[t^{n-1}]\left(\frac{t}{f(t)}\right)^n = \frac{1}{n}[t^{n-1}]\phi(t)^n.$$

The LIF can also give us the coefficients of the powers $w(t)^k$, but we can obtain a still more general result. Let $F(t) \in \mathcal{F}$ and let us consider the composition $F(w(t))$ where $w = w(t)$ is, as before, the solution to the functional equation $w = t\phi(w)$, with $\phi(w) \in \mathcal{F}_0$. For the coefficient of t^n in $F(w(t))$ we have:

$$\begin{aligned} [t^n]F(w(t)) &= [t^n]\sum_{k=0}^{\infty} F_k w(t)^k = \\ &= \sum_{k=0}^{\infty} F_k[t^n]w(t)^k = \sum_{k=0}^{\infty} F_k\frac{k}{n}[t^{n-k}]\phi(t)^n = \\ &= \frac{1}{n}[t^{n-1}]\left(\sum_{k=0}^{\infty} kF_k t^{k-1}\right)\phi(t)^n = \\ &= \frac{1}{n}[t^{n-1}]F'(t)\phi(t)^n. \end{aligned}$$

Note that $[t^0]F(w(t)) = F_0$, and this formula can be generalized to every $F(t) \in \mathcal{L}$, except for the coefficient $[t^0]F(w(t))$.

3.9 Some examples of the LIF

We found that the number b_n of binary trees with n nodes (and of other combinatorial objects

as well) satisfies the recurrence relation: $b_{n+1} = \sum_{k=0}^{n} b_k b_{n-k}$. Let us consider the f.p.s. $b(t) = \sum_{k=0}^{\infty} b_k t^k$; if we multiply the recurrence relation by t^{n+1} and sum for n from 0 to infinity, we find:

$$\sum_{n=0}^{\infty} b_{n+1} t^{n+1} = \sum_{n=0}^{\infty} t^{n+1} \left(\sum_{k=0}^{n} b_k b_{n-k} \right).$$

Since $b_0 = 1$, we can add and subtract $1 = b_0 t^0$ from the left hand member and can take t outside the summation sign in the right hand member:

$$\sum_{n=0}^{\infty} b_n t^n - 1 = t \sum_{n=0}^{\infty} \left(\sum_{k=0}^{n} b_k b_{n-k} \right) t^n.$$

In the r.h.s. we recognize a convolution and substituting $b(t)$ for the corresponding f.p.s., we obtain:

$$b(t) - 1 = t b(t)^2.$$

We are interested in evaluating $b_n = [t^n]b(t)$; let us therefore set $w = w(t) = b(t) - 1$, so that $w(t) \in \mathcal{F}_1$ and $w_n = b_n, \forall n > 0$. The previous relation becomes $w = t(1+w)^2$ and we see that the LIF can be applied (in the form relative to the functional equation) with $\phi(t) = (1+t)^2$. Therefore we have:

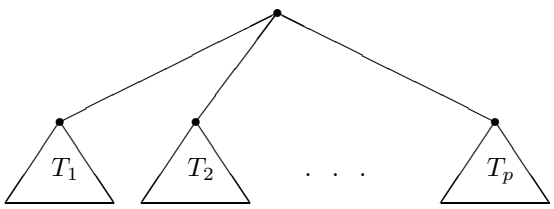
$$\begin{aligned} b_n &= [t^n]w(t) = \frac{1}{n}[t^{n-1}](1+t)^{2n} = \\ &= \frac{1}{n}\binom{2n}{n-1} = \frac{1}{n}\frac{(2n)!}{(n-1)!(n+1)!} = \\ &= \frac{1}{n+1}\frac{(2n)!}{n!n!} = \frac{1}{n+1}\binom{2n}{n}. \end{aligned}$$

As we said in the previous chapter, b_n is called the nth Catalan number and, under this name, it is often denoted by C_n. Now we have its form:

$$C_n = \frac{1}{n+1}\binom{2n}{n}$$

also valid in the case $n = 0$ when $C_0 = 1$.

In the same way we can compute the number of *p-ary trees* with n nodes. A p-ary tree is a tree in which all the nodes have arity p, except for leaves, which have arity 0. A non-empty p-ary tree can be decomposed in the following way:



which proves that $T_{n+1} = \sum T_{i_1} T_{i_2} \cdots T_{i_p}$, where the sum is extended to all the p-uples $(i_1, i_2, \ldots, i_p)$ such that $i_1 + i_2 + \cdots + i_p = n$. As before, we can multiply by t^{n+1} the two members of the recurrence relation and sum for n from 0 to infinity. We find:

$$T(t) - 1 = tT(t)^p.$$

This time we have a p degree equation, which cannot be directly solved. However, if we set $w(t) = T(t) - 1$, so that $w(t) \in \mathcal{F}_1$, we have:

$$w = t(1+w)^p$$

and the LIF gives:

$$\begin{aligned} T_n &= [t^n]w(t) = \frac{1}{n}[t^{n-1}](1+t)^{pn} = \\ &= \frac{1}{n}\binom{pn}{n-1} = \frac{1}{n}\frac{(pn)!}{(n-1)!((p-1)n+1)!} = \\ &= \frac{1}{(p-1)n+1}\binom{pn}{n} \end{aligned}$$

which generalizes the formula for the Catalan numbers.

Finally, let us find the solution of the functional equation $w = te^w$. The LIF gives:

$$w_n = \frac{1}{n}[t^{n-1}]e^{nt} = \frac{1}{n}\frac{n^{n-1}}{(n-1)!} = \frac{n^{n-1}}{n!}.$$

Therefore, the solution we are looking for is the f.p.s.:

$$w(t) = \sum_{n=1}^{\infty} \frac{n^{n-1}}{n!} t^n = t + t^2 + \frac{3}{2}t^3 + \frac{8}{3}t^4 + \frac{125}{24}t^5 + \cdots.$$

As noticed, $w(t)$ is the compositional inverse of

$$f(t) = t/\phi(t) = te^{-t} = t - t^2 + \frac{t^3}{2!} - \frac{t^4}{3!} + \frac{t^5}{4!} - \cdots.$$

It is a useful exercise to perform the necessary computations to show that $f(w(t)) = t$, for example up to the term of degree 5 or 6, and verify that $w(f(t)) = t$ as well.

3.10 Formal power series and the computer

When we are dealing with generating functions, or more in general with formal power series of any kind, we often have to perform numerical computations in order to verify some theoretical result or to experiment with actual cases. In these and other circumstances the computer can help very much with its speed and precision. Nowadays, several Computer Algebra Systems exist, which offer the possibility of

actually working with formal power series, containing formal parameters as well. The use of these tools is recommended because they can solve a doubt in a few seconds, can clarify difficult theoretical points and can give useful hints whenever we are faced with particular problems.

However, a Computer Algebra System is not always accessible or, in certain circumstances, one may desire to use less sophisticated tools. For example, programmable pocket computers are now available, which can perform quite easily the basic operations on formal power series. The aim of the present and of the following sections is to describe the main algorithms for dealing with formal power series. They can be used to program a computer or to simply understand how an existing system actually works.

The simplest way to represent a formal power series is surely by means of a vector, in which the kth component (starting from 0) is the coefficient of t^k in the power series. Obviously, the computer memory only can store a finite number of components, so an upper bound n_0 is usually given to the length of vectors and to represent power series. In other words we have:

$$\mathrm{repr}_{n_0}\left(\sum_{k=0}^{\infty} a_k t^k\right) = (a_0, a_1, \ldots, a_n) \qquad (n \le n_0)$$

Fortunately, most operations on formal power series preserve the number of significant components, so that there is little danger that a number of successive operations could reduce a finite representation to a meaningless sequence of numbers. Differentiation decreases by one the number of useful components; on the contrary, integration and multiplication by t^r, say, increase the number of significant elements, at the cost of introducing some 0 components.

The components $a_0, a_1, \ldots, a_n$ are usually real numbers, represented with the precision allowed by the particular computer. In most combinatorial applications, however, $a_0, a_1, \ldots, a_n$ are rational numbers and, with some extra effort, it is not difficult to realize rational arithmetic on a computer. It is sufficient to represent a rational number as a couple (m, n), whose intended meaning is just m/n. So we must have $m \in \mathbb{Z}, n \in \mathbb{N}$ and it is a good idea to keep m and n coprime. This can be performed by a routine *reduce* computing $p = \gcd(m, n)$ using Euclid's algorithm and then dividing both m and n by p. The operations on rational numbers are defined in the following way:

$$\begin{aligned}
(m,n) + (m',n') &= \mathrm{reduce}(mn' + m'n, nn') \\
(m,n) \times (m',n') &= \mathrm{reduce}(mm', nn') \\
(m,n)^{-1} &= (n,m) \\
(m,n)^p &= (m^p, n^p) \qquad (p \in \mathbb{N})
\end{aligned}$$

provided that (m, n) is a reduced rational number. The dimension of m and n is limited by the internal representation of integer numbers.

In order to avoid this last problem, Computer Algebra Systems usually realize an *indefinite precision* integer arithmetic. An integer number has a variable length internal representation and special routines are used to perform the basic operations. These routines can also be realized in a high level programming language (such as C or JAVA), but they can slow down too much execution time if realized on a programmable pocket computer.

3.11 The internal representation of expressions

The simple representation of a formal power series by a vector of real, or rational, components will be used in the next sections to explain the main algorithms for formal power series operations. However, it is surely not the best way to represent power series and becomes completely useless when, for example, the coefficients depend on some formal parameter. In other words, our representation only can deal with purely numerical formal power series.

Because of that, Computer Algebra Systems use a more sophisticated internal representation. In fact, power series are simply a particular case of a general mathematical expression. The aim of the present section is to give a rough idea of how an expression can be represented in the computer memory.

In general, an expression consists in *operators* and *operands*. For example, in $a + \sqrt{3}$, the operators are $+$ and $\sqrt{\cdot}$, and the operands are a and 3. Every operator has its own *arity* or *adicity*, i.e., the number of operands on which it acts. The adicity of the sum $+$ is two, because it acts on its two terms; the adicity of the square root $\sqrt{\cdot}$ is one, because it acts on a single term. Operands can be numbers (and it is important that the nature of the number be specified, i.e., if it is a natural, an integer, a rational, a real or a complex number) or can be a formal parameter, as a. Obviously, if an operator acts on numerical operands, it can be executed giving a numerical result. But if any of its operands is a formal parameter, the result is a formal expression, which may perhaps be simplified but cannot be evaluated to a numerical result.

An expression can always be transposed into a "tree", the internal nodes of which correspond to operators and whose leaves correspond to operands. The simple tree for the previous expression is given in Figure 3.1. Each operator has as many branches as its adicity is and a simple visit to the tree can perform its evaluation, that is it can execute all the operators

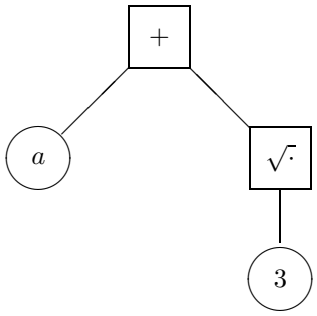


Figure 3.1: The tree for a simple expression

when only numerical operands are attached to them. Simplification is a rather complicated matter and it is not quite clear what a "simple" expression is. For example, which is simpler between $(a+1)(a+2)$ and a^2+3a+2? It is easily seen that there are occasions in which either expression can be considered "simpler". Therefore, most Computer Algebra Systems provide both a general "simplification" routine together with a series of more specific programs performing some specific simplifying tasks, such as expanding parenthetized expressions or collecting like factors.

In the computer memory an expression is represented by its tree, which is called the *tree* or *list representation* of the expression. The representation of the formal power series $\sum_{k=0}^{n} a_k t^k$ is shown in Figure 3.2. This representation is very convenient and when some coefficients $a_1, a_2, \ldots, a_n$ depend on a formal parameter p nothing is changed, at least conceptually. In fact, where we have drawn a leaf a_j, we simply have a more complex tree representing the expression for a_j.

The reader can develop computer programs for dealing with this representation of formal power series. It should be clear that another important point of this approach is that no limitation is given to the length of expressions. A clever and dynamic use of the storage solves every problem without increasing the complexity of the corresponding programs.

3.12 Basic operations of formal power series

We are now considering the vector representation of formal power series:

$$\mathrm{repr}_{n_0}\left(\sum_{k=0}^{\infty} a_k t^k\right) = (a_0, a_1, \ldots, a_n) \qquad n \le n_0.$$

The sum of the formal power series is defined in an obvious way:

$$(a_0, a_1, \ldots, a_n) + (b_0, b_1, \ldots, b_m) = (c_0, c_1, \ldots, c_r)$$

where $c_i = a_i + b_i$ for every $0 \le i \le r$, and $r = \min(n, m)$. In a similar way the Cauchy product is defined:

$$(a_0, a_1, \ldots, a_n) \times (b_0, b_1, \ldots, b_m) = (c_0, c_1, \ldots, c_r)$$

where $c_k = \sum_{j=0}^{k} a_j b_{k-j}$ for every $0 \le k \le r$. Here r is defined as $r = \min(n + p_B, m + p_A)$, if p_A is the first index for which $a_{p_A} \neq 0$ and p_B is the first index for which $b_{p_B} \neq 0$. We point out that the time complexity for the sum is $O(r)$ and the time complexity for the product is $O(r^2)$.

Subtraction is similar to addition and does not require any particular comment. Before discussing division, let us consider the operation of rising a formal power series to a power $\alpha \in \mathbb{R}$. This includes the inversion of a power series ($\alpha = -1$) and therefore division as well.

First of all we observe that whenever $\alpha \in \mathbb{N}$, $f(t)^\alpha$ can be reduced to that case $(1 + g(t))^\alpha$, where $g(t) \notin \mathcal{F}_0$. In fact we have:

$$\begin{aligned} f(t) &= f_h t^h + f_{h+1} t^{h+1} + f_{h+2} t^{h+2} + \cdots = \\ &= f_h t^h \left(1 + \frac{f_{h+1}}{f_h} t + \frac{f_{h+2}}{f_h} t^2 + \cdots\right) \end{aligned}$$

and therefore:

$$f(t)^\alpha = f_h^\alpha t^{\alpha h} \left(1 + \frac{f_{h+1}}{f_h} t + \frac{f_{h+2}}{f_h} t^2 + \cdots\right)^\alpha.$$

On the contrary, when $\alpha \notin \mathbb{N}$, $f(t)^\alpha$ only can be performed if $f(t) \in \mathcal{F}_0$. In that case we have:

$$\begin{aligned} f(t)^\alpha &= \left(f_0 + f_1 t + f_2 t^2 + f_3 t^3 + \cdots\right)^\alpha = \\ &= f_0^\alpha \left(1 + \frac{f_1}{f_0} t + \frac{f_2}{f_0} t^2 + \frac{f_3}{f_0} t^3 + \cdots\right)^\alpha; \end{aligned}$$

note that in this case if $f_0 \neq 1$, usually f_0^α is not rational. In any case, we are always reduced to compute $(1 + g(t))^\alpha$ and since:

$$(1 + g(t))^\alpha = \sum_{k=0}^{\infty} \binom{\alpha}{k} g(t)^k \tag{3.12.1}$$

if the coefficients in $g(t)$ are rational numbers, also the coefficients in $(1 + g(t))^\alpha$ are, provided $\alpha \in \mathbb{Q}$. These considerations are to be remembered if the operation is realized in some special environment, as described in the previous section.

Whatever $\alpha \in \mathbb{R}$ is, the exponents involved in the right hand member of (3.12.1) are all positive integer numbers. Therefore, powers can be realized as successive multiplications or Cauchy products. This gives a straight-forward method for performing $(1 + g(t))^\alpha$, but it is easily seen to take a time in the order of $O(r^3)$, since it requires r products each executed in time $O(r^2)$. Fortunately, however,

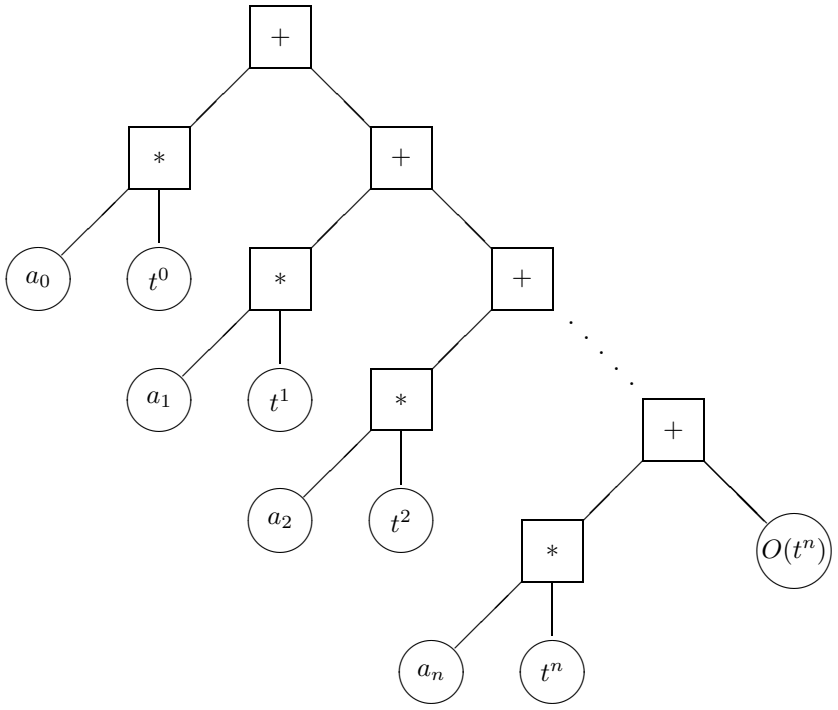


Figure 3.2: The tree for a formal power series

J. C. P. Miller has devised an algorithm allowing to perform $(1 + g(t))^\alpha$ in time $O(r^2)$. In fact, let us write $h(t) = a(t)^\alpha$, where $a(t)$ is any formal power series with $a_0 = 1$. By differentiating, we obtain $h'(t) = \alpha a(t)^{\alpha-1}a'(t)$, or, by multiplying everything by $a(t)$, $a(t)h'(t) = \alpha h(t)a'(t)$. Therefore, by extracting the coefficient of t^{n-1} we find:

$$\sum_{k=0}^{n-1} a_k(n-k)h_{n-k} = \alpha \sum_{k=0}^{n-1}(k+1)a_{k+1}h_{n-k-1}$$

We now isolate the term with $k = 0$ in the left hand member and the term having $k = n - 1$ in the right hand member ($a_0 = 1$ by hypothesis):

$$nh_n + \sum_{k=1}^{n-1} a_k(n-k)h_{n-k} = \alpha n a_n + \sum_{k=1}^{n-1} \alpha k a_k h_{n-k}$$

(in the last sum we performed the change of variable $k \to k - 1$, in order to have the same indices as in the left hand member). We now have an expression for h_n only depending on $(a_1, a_2, \ldots, a_n)$ and $(h_1, h_2, \ldots, h_{n-1})$:

$$\begin{aligned} h_n &= \alpha a_n + \frac{1}{n}\sum_{k=1}^{n-1}\left((\alpha+1)k - n\right)a_k h_{n-k} = \\ &= \alpha a_n + \sum_{k=1}^{n-1}\left(\frac{(\alpha+1)k}{n} - 1\right)a_k h_{n-k} \end{aligned}$$

The computation is now straight-forward. We begin by setting $h_0 = 1$, and then we successively compute $h_1, h_2, \ldots, h_r$ ($r = n$, if n is the number of terms in $(a_1, a_2, \ldots, a_n)$). The evaluation of h_k requires a number of operations in the order $O(k)$, and therefore the whole procedure works in time $O(r^2)$, as desired.

The inverse of a series, i.e., $(1+g(t))^{-1}$, is obtained by setting $\alpha = -1$. It is worth noting that the previous formula becomes:

$$h_k = -a_k - \sum_{j=1}^{k-1} a_j h_{k-j} = -\sum_{j=1}^{k} a_j h_{k-j} \quad (h_0 = 1)$$

and can be used to prove properties of the inverse of a power series. As a simple example, the reader can show that the coefficients in $(1 - t)^{-1}$ are all 1.

3.13 Logarithm and exponential

The idea of Miller can be applied to other operations on formal power series. In the present section we wish to use it to perform the (natural) logarithm and the exponentiation of a series. Let us begin with the logarithm and try to compute $\ln(1 + g(t))$. As we know, there is a direct way to perform this operation, i.e.:

$$\ln(1 + g(t)) = \sum_{k=1}^{\infty} \frac{1}{k} g(t)^k$$

and this formula only requires a series of successive products. As for the operation of rising to a power, the procedure needs a time in the order of $O(r^3)$, and it is worth considering an alternative approach. In fact, if we set $h(t) = \ln(1+g(t))$, by differentiating we obtain $h'(t) = g'(t)/(1+g(t))$, or $h'(t) = g'(t) - h'(t)g(t)$. We can now extract the coefficient of t^{k-1} and obtain:

$$kh_k = kg_k - \sum_{j=0}^{k-1}(k-j)h_{k-j}g_j$$

However, $g_0 = 0$ by hypothesis, and therefore we have an expression relating h_k to $(g_1, g_2, \ldots, g_k)$ and to $(h_1, h_2, \ldots, h_{k-1})$:

$$\begin{aligned} h_k &= g_k - \frac{1}{k}\sum_{j=1}^{k-1}(k-j)h_{k-j}g_j = \\ &= g_k - \sum_{j=1}^{k-1}\left(1 - \frac{j}{k}\right)h_{k-j}g_j \end{aligned}$$

A program to perform the logarithm of a formal power series $1+g(t)$ begins by setting $h_0 = 0$ and then proceeds computing $h_1, h_2, \ldots, h_r$ if r is the number of significant terms in $g(t)$. The total time is clearly in the order of $O(r^2)$.

A similar technique can be applied to the computation of $\exp(g(t))$ provided that $g(t) \notin \mathcal{F}_0$. If $g(t) \in \mathcal{F}_0$, i.e., $g(t) = g_0 + g_1 t + g_2 t^2 + \cdots$, we have $\exp(g_0 + g_1 t + g_2 t^2 + \cdots) = e^{g_0}\exp(g_1 t + g_2 t^2 + \cdots)$. In this way we are reduced to the previous case, but we no longer have rational coefficients when $g(t) \in \mathbb{Q}[\![t]\!]$.

By differentiating the identity $h(t) = \exp(g(t))$ we obtain $h'(t) = g'(t)\exp(g(t)) = g'(t)h(t)$. We extract the coefficient of t^{k-1}:

$$kh_k = \sum_{j=0}^{k-1}(j+1)g_{j+1}h_{k-j-1} = \sum_{j=1}^{k} jg_j h_{k-j}$$

This formula allows us to compute h_k in terms of $(g_1, g_2, \ldots, g_k)$ and $(h_0 = 1, h_1, h_2, \ldots, h_{k-1})$. A program performing exponentiation can be easily written by defining $h_0 = 1$ and successively evaluating $h_1, h_2, \ldots, h_r$. if r is the number of significant terms in $g(t)$. Time complexity is obviously $O(r^2)$.

Unfortunately, a similar trick does not work for series composition. To compute $f(g(t))$, when $g(t) \notin \mathcal{F}_0$, we have to resort to the defining formula:

$$f(g(t)) = \sum_{k=0}^{\infty} f_k g(t)^k$$

This requires the successive computation of the integer powers of $g(t)$, which can be performed by repeated applications of the Cauchy product. The execution time is in the order of $O(r^3)$, if r is the minimal number of significant terms in $f(t)$ and $g(t)$, respectively.

We conclude this section by sketching the obvious algorithms to compute differentiation and integration of a formal power series $f(t) = f_0 + f_1 t + f_2 t^2 + f_3 t^3 + \cdots$. If $h(t) = f'(t)$, we have:

$$h_k = (k+1)f_{k+1}$$

and therefore the number of significant terms is reduced by 1. Conversely, if $h(t) = \int_0^t f(\tau)d\tau$, we have:

$$h_k = \frac{1}{k}f_{k-1}$$

and $h_0 = 0$; consequently the number of significant terms is increased by 1.

Chapter 4

Generating Functions

4.1 General Rules

Let us consider a sequence of numbers $F = (f_0, f_1, f_2, \ldots) = (f_k)_{k\in N}$; the *(ordinary) generating function* for the sequence F is defined as $f(t) = f_0 + f_1 t + f_2 t^2 + \cdots$, where the indeterminate t is arbitrary. Given the sequence $(f_k)_{k\in N}$, we introduce the *generating function operator* $\mathcal{G}$, which applied to $(f_k)_{k\in N}$ produces the ordinary generating function for the sequence, i.e., $\mathcal{G}(f_k)_{k\in N} = f(t)$. In this expression t is a bound variable, and a more accurate notation would be $\mathcal{G}_t\,(f_k)_{k\in N} = f(t)$. This notation is essential when $(f_k)_{k\in N}$ depends on some parameter or when we consider multivariate generating functions. In this latter case, for example, we should write $\mathcal{G}_{t,w}\,(f_{n,k})_{n,k\in N} = f(t,w)$ to indicate the fact that $f_{n,k}$ in the double sequence becomes the coefficient of $t^n w^k$ in the function $f(t,w)$. However, whenever no ambiguity can arise, we will use the notation $\mathcal{G}(f_k) = f(t)$, understanding also the binding for the variable k. For the sake of completeness, we also define the *exponential generating function* of the sequence $(f_0, f_1, f_2, \ldots)$ as:

$$\mathcal{E}(f_k) = \mathcal{G}\left(\frac{f_k}{k!}\right) = \sum_{k=0}^{\infty} f_k \frac{t^k}{k!}.$$

The operator $\mathcal{G}$ is clearly linear. The function $f(t)$ can be shifted or differentiated. Two functions $f(t)$ and $g(t)$ can be multiplied and composed. This leads to the properties for the operator $\mathcal{G}$ listed in Table 4.1. Note that formula $(G5)$ requires $g_0 = 0$. The first five formulas are easily verified by using the intended interpretation of the operator $\mathcal{G}$; the last formula can be proved by means of the LIF, in the form relative to the composition $F(w(t))$. In fact we have:

$$\begin{aligned}[t^n]F(t)\phi(t)^n &= [t^{n-1}]\frac{F(t)}{t}\phi(t)^n = \\ &= n[t^n]\left[\int \frac{F(y)}{y}dy \;\middle|\; y = w(t)\right];\end{aligned}$$

in the last passage we applied backwards the formula:

$$[t^n]F(w(t)) = \frac{1}{n}[t^{n-1}]F'(t)\phi(t)^n \quad (w = t\phi(w))$$

and therefore $w = w(t) \in \mathcal{F}_1$ is the unique solution of the functional equation $w = t\phi(w)$. By now applying the rule of differentiation for the "coefficient of" operator, we can go on:

$$\begin{aligned}&[t^n]F(t)\phi(t)^n \\ &= [t^{n-1}]\frac{d}{dt}\left[\int \frac{F(y)}{y}dy \;\middle|\; y = w(t)\right] = \\ &= [t^{n-1}]\left[\frac{F(w)}{w} \;\middle|\; w = t\phi(w)\right]\frac{dw}{dt}.\end{aligned}$$

We have applied the chain rule for differentiation, and from $w = t\phi(w)$ we have:

$$\frac{dw}{dt} = \phi(w) + t\left[\frac{d\phi}{dw} \;\middle|\; w = t\phi(w)\right]\frac{dw}{dt}.$$

We can therefore compute the derivative of $w(t)$:

$$\frac{dw}{dt} = \left[\frac{\phi(w)}{1 - t\phi'(w)} \;\middle|\; w = t\phi(w)\right]$$

where $\phi'(w)$ denotes the derivative of $\phi(w)$ with respect to w. We can substitute this expression in the formula above and observe that $w/\phi(w) = t$ can be taken outside of the substitution symbol:

$$\begin{aligned}&[t^n]F(t)\phi(t)^n = \\ &= [t^{n-1}]\left[\frac{F(w)}{w}\frac{\phi(w)}{1 - t\phi'(w)} \;\middle|\; w = t\phi(w)\right] = \\ &= [t^{n-1}]\frac{1}{t}\left[\frac{F(w)}{1 - t\phi'(w)} \;\middle|\; w = t\phi(w)\right] = \\ &= [t^n]\left[\frac{F(w)}{1 - t\phi'(w)} \;\middle|\; w = t\phi(w)\right]\end{aligned}$$

which is our diagonalization rule $(G6)$.

The name "diagonalization" is due to the fact that if we imagine the coefficients of $F(t)\phi(t)^n$ as constituting the row n in an infinite matrix, then

linearity	$\mathcal{G}(\alpha f_k + \beta g_k)$	$=$	$\alpha\mathcal{G}(f_k) + \beta\mathcal{G}(g_k)$	$(G1)$
shifting	$\mathcal{G}(f_{k+1})$	$=$	$\frac{(\mathcal{G}(f_k) - f_0)}{t}$	$(G2)$
differentiation	$\mathcal{G}(kf_k)$	$=$	$tD\mathcal{G}(f_k)$	$(G3)$
convolution	$\mathcal{G}\left(\sum_{k=0}^{n} f_k g_{n-k}\right)$	$=$	$\mathcal{G}(f_k)\cdot\mathcal{G}(g_k)$	$(G4)$
composition	$\sum_{n=0}^{\infty} f_n \left(\mathcal{G}(g_k)\right)^n$	$=$	$\mathcal{G}(f_k)\circ\mathcal{G}(g_k)$	$(G5)$
diagonalisation	$\mathcal{G}([t^n]F(t)\phi(t)^n)$	$=$	$\left[\frac{F(w)}{1-t\phi'(w)} \,\middle|\, w = t\phi(w)\right]$	$(G6)$

Table 4.1: The rules for the generating function operator

$[t^n]F(t)\phi(t)^n$ are just the elements in the main diagonal of this array.

The rules $(G1) - (G6)$ can also be assumed as axioms of a theory of generating functions and used to derive general theorems as well as specific functions for particular sequences. In the next sections, we will prove a number of properties of the generating function operator. The proofs rely on the following fundamental *principle of identity*:

Given two sequences $(f_k)_{k\in N}$ and $(g_k)_{k\in N}$, then $\mathcal{G}(f_k) = \mathcal{G}(g_k)$ if and only if for every $k \in \mathbb{N}$ $f_k = g_k$.

The principle is rather obvious from the very definition of the concept of generating functions; however, it is important, because it states the condition under which we can pass from an identity about elements to the corresponding identity about generating functions. It is sufficient that the two sequences do not agree by a single element (e.g., the first one) and we cannot infer the equality of generating functions.

4.2 Some Theorems on Generating Functions

We are now going to prove a series of properties of generating functions.

Theorem 4.2.1 *Let $f(t) = \mathcal{G}(f_k)$ be the generating function of the sequence $(f_k)_{k\in N}$, then*

$$\mathcal{G}(f_{k+2}) = \frac{\mathcal{G}(f_k) - f_0 - f_1 t}{t^2} \tag{4.2.1}$$

Proof: Let $g_k = f_{k+1}$; by $(G2)$, $\mathcal{G}(g_k) = (\mathcal{G}(f_k) - f_0)/t$. Since $g_0 = f_1$, we have:

$$\mathcal{G}(f_{k+2}) = \mathcal{G}(g_{k+1}) = \frac{\mathcal{G}(g_k) - g_0}{t} = \frac{\mathcal{G}(f_k) - f_0 - f_1 t}{t^2}$$

∎

By mathematical induction this result can be generalized to:

Theorem 4.2.2 *Let $f(t) = \mathcal{G}(f_k)$ be as above; then:*

$$\mathcal{G}(f_{k+j}) = \frac{\mathcal{G}(f_k) - f_0 - f_1 t - \cdots - f_{j-1}t^{j-1}}{t^j} \tag{4.2.2}$$

If we consider right instead of left shifting we have to be more careful:

Theorem 4.2.3 *Let $f(t) = \mathcal{G}(f_k)$ be as above, then:*

$$\mathcal{G}(f_{k-j}) = t^j \mathcal{G}(f_k) \tag{4.2.3}$$

Proof: We have $\mathcal{G}(f_k) = \mathcal{G}\big(f_{(k-1)+1}\big) = t^{-1}\left(\mathcal{G}(f_{n-1}) - f_{-1}\right)$ where f_{-1} is the coefficient of t^{-1} in $f(t)$. If $f(t) \in \mathcal{F}$, $f_{-1} = 0$ and $\mathcal{G}(f_{k-1}) = t\mathcal{G}(f_k)$. The theorem then follows by mathematical induction. ∎

Property $(G3)$ can be generalized in several ways:

Theorem 4.2.4 *Let $f(t) = \mathcal{G}(f_k)$ be as above; then:*

$$\mathcal{G}((k+1)f_{k+1}) = D\mathcal{G}(f_k) \tag{4.2.4}$$

Proof: If we set $g_k = kf_k$, we obtain:

$$\begin{aligned}\mathcal{G}((k+1)f_{k+1}) &= \mathcal{G}(g_{k+1}) = \\ &= t^{-1}\left(\mathcal{G}(kf_k) - 0f_0\right) \overset{(G2)}{=} \\ &= t^{-1}tD\mathcal{G}(f_k) = D\mathcal{G}(f_k)\,.\end{aligned}$$

∎

Theorem 4.2.5 *Let $f(t) = \mathcal{G}(f_k)$ be as above; then:*

$$\mathcal{G}\big(k^2 f_k\big) = tD\mathcal{G}(f_k) + t^2 D^2\mathcal{G}(f_k) \tag{4.2.5}$$

This can be further generalized:

Theorem 4.2.6 *Let $f(t) = \mathcal{G}(f_k)$ be as above; then:*

$$\mathcal{G}\big(k^j f_k\big) = \mathcal{S}_j(tD)\mathcal{G}(f_k) \tag{4.2.6}$$

where $\mathcal{S}_j(w) = \sum_{r=1}^{j} \left\{ {j \atop r} \right\} w^r$ is the jth Stirling polynomial of the second kind (see Section 2.11).

Proof: Formula (4.2.6) is to be understood in the operator sense; so, for example, being $\mathcal{S}_3(w) = w + 3w^2 + w^3$, we have:

$$\mathcal{G}\big(k^3 f_k\big) = tD\mathcal{G}(f_k) + 3t^2D^2\mathcal{G}(f_k) + t^3D^3\mathcal{G}(f_k)\,.$$

The proof proceeds by induction, as $(G3)$ and (4.2.5) are the first two instances. Now:

$$\mathcal{G}\big(k^{j+1} f_k\big) = \mathcal{G}\big(k(k^j) f_k\big) = tD\mathcal{G}\big(k^j f_k\big)$$

that is:

$$\mathcal{S}_{j+1}(tD) = tD\mathcal{S}_j(tD) = tD\sum_{r=1}^{j} S(j,r)t^r D^r =$$

$$= \sum_{r=1}^{j} S(j,r)rt^r D^r + \sum_{r=1}^{j} S(j,r)t^{r+1}D^{r+1}.$$

By equating like coefficients we find $S(j+1,r) = rS(j,r) + S(j,r-1)$, which is the classical recurrence for the Stirling numbers of the second kind. Since initial conditions also coincide, we can conclude $S(j,r) = \left\{ {j \atop r} \right\}$. ∎

For the falling factorial $k^{\underline{r}} = k(k-1)\cdots(k-r+1)$ we have a simpler formula, the proof of which is immediate:

Theorem 4.2.7 *Let $f(t) = \mathcal{G}(f_k)$ be as above; then:*

$$\mathcal{G}(k^{\underline{r}} f_k) = t^r D^r \mathcal{G}(f_k) \tag{4.2.7}$$

Let us now come to integration:

Theorem 4.2.8 *Let $f(t) = \mathcal{G}(f_k)$ be as above and let us define $g_k = f_k/k, \forall k \neq 0$ and $g_0 = 0$; then:*

$$\mathcal{G}\left(\frac{1}{k} f_k\right) = \mathcal{G}(g_k) = \int_0^t (\mathcal{G}(f_k) - f_0)\,\frac{dz}{z} \tag{4.2.8}$$

Proof: Clearly, $kg_k = f_k$, except for $k = 0$. Hence we have $\mathcal{G}(kg_k) = \mathcal{G}(f_k) - f_0$. By using $(G3)$, we find $tD\mathcal{G}(g_k) = \mathcal{G}(f_k) - f_0$, from which (4.2.8) follows by integration and the condition $g_0 = 0$. ∎

A more classical formula is:

Theorem 4.2.9 *Let $f(t) = \mathcal{G}(f_k)$ be as above or, equivalently, let $f(t)$ be a f.p.s. but not a f.L.s.; then:*

$$\mathcal{G}\left(\frac{1}{k+1} f_k\right) =$$

$$= \frac{1}{t}\int_0^t \mathcal{G}(f_k)\,dz = \frac{1}{t}\int_0^t f(z)\,dz \tag{4.2.9}$$

Proof: Let us consider the sequence $(g_k)_{k\in N}$, where $g_{k+1} = f_k$ and $g_0 = 0$. So we have: $\mathcal{G}(g_{k+1}) = \mathcal{G}(f_k) = t^{-1}(\mathcal{G}(g_k) - g_0)$. Finally:

$$\mathcal{G}\left(\frac{1}{k+1} f_k\right) = \mathcal{G}\left(\frac{1}{k+1} g_{k+1}\right) = \frac{1}{t}\mathcal{G}\left(\frac{1}{k} g_k\right) =$$

$$= \frac{1}{t}\int_0^t (\mathcal{G}(g_k) - g_0)\,\frac{dz}{z} = \frac{1}{t}\int_0^t \mathcal{G}(f_k)\,dz$$

∎

In the following theorems, $f(t)$ will always denote the generating function $\mathcal{G}(f_k)$.

Theorem 4.2.10 *Let $f(t) = \mathcal{G}(f_k)$ denote the generating function of the sequence $(f_k)_{k\in N}$; then:*

$$\mathcal{G}\big(p^k f_k\big) = f(pt) \tag{4.2.10}$$

Proof: By setting $g(t) = pt$ in $(G5)$ we have: $f(pt) = \sum_{n=0}^{\infty} f_n (pt)^n = \sum_{n=0}^{\infty} p^n f_n t^n = \mathcal{G}\big(p^k f_k\big)$ ∎

In particular, for $p = -1$ we have: $\mathcal{G}\big((-1)^k f_k\big) = f(-t)$.

4.3 More advanced results

The results obtained till now can be considered as simple generalizations of the axioms. They are very useful and will be used in many circumstances. However, we can also obtain more advanced results, concerning sequences derived from a given sequence by manipulating its elements in various ways. For example, let us begin by proving the well-known bisection formulas:

Theorem 4.3.1 *Let $f(t) = \mathcal{G}(f_k)$ denote the generating function of the sequence $(f_k)_{k\in N}$; then:*

$$\mathcal{G}(f_{2k}) = \frac{f(\sqrt{t}) + f(-\sqrt{t})}{2} \tag{4.3.1}$$

$$\mathcal{G}(f_{2k+1}) = \frac{f(\sqrt{t}) - f(-\sqrt{t})}{2\sqrt{t}} \tag{4.3.2}$$

where $\sqrt{t}$ is a symbol with the property that $(\sqrt{t})^2 = t$.

Proof: By $(G5)$ we have:

$$\frac{f(\sqrt{t}) + f(-\sqrt{t})}{2} =$$

$$= \frac{\sum_{n=0}^{\infty} f_n \sqrt{t}^n + \sum_{n=0}^{\infty} f_n (-\sqrt{t})^n}{2} =$$

$$= \sum_{n=0}^{\infty} f_n \frac{(\sqrt{t})^n + (-\sqrt{t})^n}{2}$$

For n odd $(\sqrt{t})^n + (-\sqrt{t})^n = 0$; hence, by setting $n = 2k$, we have:

$$\sum_{k=0}^{\infty} f_{2k}(t^k + t^k)/2 = \sum_{k=0}^{\infty} f_{2k} t^k = \mathcal{G}(f_{2k})\,.$$

The proof of the second formula is analogous. ∎

The following proof is typical and introduces the use of ordinary differential equations in the calculus of generating functions:

Theorem 4.3.2 *Let $f(t) = \mathcal{G}(f_k)$ denote the generating function of the sequence $(f_k)_{k\in N}$; then:*

$$\mathcal{G}\left(\frac{f_k}{2k+1}\right) = \frac{1}{2\sqrt{t}}\int_0^t \frac{\mathcal{G}(f_k)}{\sqrt{z}}\,dz \tag{4.3.3}$$

Proof: Let us set $g_k = f_k/(2k+1)$, or $2kg_k+g_k = f_k$. If $g(t) = \mathcal{G}(g_k)$, by applying $(G3)$ we have the differential equation $2tg'(t) + g(t) = f(t)$, whose solution having $g(0) = f_0$ is just formula (4.3.3). ∎

We conclude with two general theorems on sums:

Theorem 4.3.3 (Partial Sum Theorem) *Let $f(t) = \mathcal{G}(f_k)$ denote the generating function of the sequence $(f_k)_{k\in N}$; then:*

$$\mathcal{G}\left(\sum_{k=0}^{n} f_k\right) = \frac{1}{1-t}\mathcal{G}(f_k) \tag{4.3.4}$$

Proof: If we set $s_n = \sum_{k=0}^{n} f_k$, then we have $s_{n+1} = s_n + f_{n+1}$ for every $n \in \mathbb{N}$ and we can apply the operator $\mathcal{G}$ to both members: $\mathcal{G}(s_{n+1}) = \mathcal{G}(s_n)+\mathcal{G}(f_{n+1})$, i.e.:

$$\frac{\mathcal{G}(s_n) - s_0}{t} = \mathcal{G}(s_n) + \frac{\mathcal{G}(f_n) - f_0}{t}$$

Since $s_0 = f_0$, we find $\mathcal{G}(s_n) = t\mathcal{G}(s_n) + \mathcal{G}(f_n)$ and from this (4.3.4) follows directly. ∎

The following result is known as *Euler transformation*:

Theorem 4.3.4 *Let $f(t) = \mathcal{G}(f_k)$ denote the generating function of the sequence $(f_k)_{k\in N}$; then:*

$$\mathcal{G}\left(\sum_{k=0}^{n}\binom{n}{k}f_k\right) = \frac{1}{1-t}f\left(\frac{t}{1-t}\right) \tag{4.3.5}$$

Proof: By well-known properties of binomial coefficients we have:

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

and this is the coefficient of t^{n-k} in $(1-t)^{-k-1}$. We now observe that the sum in (4.3.5) can be extended to infinity, and by $(G5)$ we have:

$$\begin{aligned}\sum_{k=0}^{n}\binom{n}{k}f_k &= \sum_{k=0}^{n}\binom{-k-1}{n-k}(-1)^{n-k}f_k = \\ &= \sum_{k=0}^{\infty}[t^{n-k}](1-t)^{-k-1}[y^k]f(y) = \\ &= [t^n]\frac{1}{1-t}\sum_{k=0}^{\infty}[y^k]f(y)\left(\frac{t}{1-t}\right)^k = \\ &= [t^n]\frac{1}{1-t}f\left(\frac{t}{1-t}\right).\end{aligned}$$

Since the last expression does not depend on n, it represents the generating function of the sum. ∎

We observe explicitly that by $(K4)$ we have: $\sum_{k=0}^{n}\binom{n}{k}f_k = [t^n](1+t)^n f(t)$, but this expression does not represent a generating function because it depends on n. The Euler transformation can be generalized in several ways, as we shall see when dealing with Riordan arrays.

4.4 Common Generating Functions

The aim of the present section is to derive the most common generating functions by using the apparatus of the previous sections. As a first example, let us consider the constant sequence $F = (1, 1, 1, \ldots)$, for which we have $f_{k+1} = f_k$ for every $k \in \mathbb{N}$. By applying the principle of identity, we find: $\mathcal{G}(f_{k+1}) = \mathcal{G}(f_k)$, that is by $(G2)$: $\mathcal{G}(f_k) - f_0 = t\mathcal{G}(f_k)$. Since $f_0 = 1$, we have immediately:

$$\mathcal{G}(1) = \frac{1}{1-t}$$

For any constant sequence $F = (c, c, c, \ldots)$, by $(G1)$ we find that $\mathcal{G}(c) = c(1-t)^{-1}$. Similarly, by using the basic rules and the theorems of the previous sections we have:

$$\begin{aligned}\mathcal{G}(n) &= \mathcal{G}(n\cdot 1) = tD\frac{1}{1-t} = \frac{t}{(1-t)^2}\\ \mathcal{G}(n^2) &= tD\mathcal{G}(n) = tD\frac{1}{(1-t)^2} = \frac{t+t^2}{(1-t)^3}\\ \mathcal{G}((-1)^n) &= \mathcal{G}(1)\circ(-t) = \frac{1}{1+t}\\ \mathcal{G}\left(\frac{1}{n}\right) &= \mathcal{G}\left(\frac{1}{n}\cdot 1\right) = \int_0^t\left(\frac{1}{1-z}-1\right)\frac{dz}{z} = \\ &= \int_0^t\frac{dz}{1-z} = \ln\frac{1}{1-t}\\ \mathcal{G}(H_n) &= \mathcal{G}\left(\sum_{k=0}^{n}\frac{1}{k}\right) = \frac{1}{1-t}\mathcal{G}\left(\frac{1}{n}\right) = \\ &= \frac{1}{1-t}\ln\frac{1}{1-t}\end{aligned}$$

where H_n is the nth harmonic number. Other generating functions can be obtained from the previous formulas:

$$\mathcal{G}(nH_n) = tD\frac{1}{1-t}\ln\frac{1}{1-t} =$$

$$\begin{aligned} &= \frac{t}{(1-t)^2}\left(\ln\frac{1}{1-t}+1\right) \\ \mathcal{G}\left(\frac{1}{n+1}H_n\right) &= \frac{1}{t}\int_0^t \frac{1}{1-z}\ln\frac{1}{1-z}\,dz = \\ &= \frac{1}{2t}\left(\ln\frac{1}{1-t}\right)^2 \\ \mathcal{G}(\delta_{0,n}) &= \mathcal{G}(1)-t\mathcal{G}(1)=\frac{1-t}{1-t}=1 \end{aligned}$$

where $\delta_{n,m}$ is the Kronecker's delta. This last relation can be readily generalized to $\mathcal{G}(\delta_{n,m}) = t^m$. An interesting example is given by $\mathcal{G}\left(\frac{1}{n(n+1)}\right)$. Since $\frac{1}{n(n+1)} = \frac{1}{n} - \frac{1}{n+1}$, it is tempting to apply the operator $\mathcal{G}$ to both members. However, this relation is not valid for $n = 0$. In order to apply the principle of identity, we must define:

$$\frac{1}{n(n+1)} = \frac{1}{n} - \frac{1}{n+1} + \delta_{n,0}$$

in accordance with the fact that the first element of the sequence is zero. We thus arrive to the correct generating function:

$$\mathcal{G}\left(\frac{1}{n(n+1)}\right) = 1 - \frac{1-t}{t}\ln\frac{1}{1-t}$$

Let us now come to binomial coefficients. In order to find $\mathcal{G}\left(\binom{p}{k}\right)$ we observe that, from the definition:

$$\begin{aligned} \binom{p}{k+1} &= \frac{p(p-1)\cdots(p-k+1)(p-k)}{(k+1)!} = \\ &= \frac{p-k}{k+1}\binom{p}{k}. \end{aligned}$$

Hence, by denoting $\binom{p}{k}$ as f_k, we have: $\mathcal{G}((k+1)f_{k+1}) = \mathcal{G}((p-k)f_k) = p\mathcal{G}(f_k) - \mathcal{G}(kf_k)$. By applying (4.2.4) and $(G3)$ we have $D\mathcal{G}(f_k) = p\mathcal{G}(f_k) - tD\mathcal{G}(f_k)$, i.e., the differential equation $f'(t) = pf(t) - tf'(t)$. By separating the variables and integrating, we find $\ln f(t) = p\ln(1+t) + c$, or $f(t) = c(1+t)^p$. For $t = 0$ we should have $f(0) = \binom{p}{0} = 1$, and this implies $c = 1$. Consequently:

$$\mathcal{G}\left(\binom{p}{k}\right) = (1+t)^p \qquad p \in \mathbb{R}$$

We are now in a position to derive the recurrence relation for binomial coefficients. By using $(K1) \div (K5)$ we find easily:

$$\begin{aligned} \binom{p}{k} &= [t^k](1+t)^p = [t^k](1+t)(1+t)^{p-1} = \\ &= [t^k](1+t)^{p-1} + [t^{k-1}](1+t)^{p-1} = \\ &= \binom{p-1}{k} + \binom{p-1}{k-1}. \end{aligned}$$

By $(K4)$, we have the well-known *Vandermonde convolution*:

$$\begin{aligned} \binom{m+p}{n} &= [t^n](1+t)^{m+p} = \\ &= [t^n](1+t)^m(1+t)^p = \\ &= \sum_{k=0}^{n}\binom{m}{k}\binom{p}{n-k} \end{aligned}$$

which, for $m = p = n$ becomes $\sum_{k=0}^{n}\binom{n}{k}^2 = \binom{2n}{n}$.

We can also find $\mathcal{G}\left(\binom{k}{p}\right)$, where p is a parameter. The derivation is purely algebraic and makes use of the generating functions already found and of various properties considered in the previous section:

$$\begin{aligned} \mathcal{G}\left(\binom{k}{p}\right) &= \mathcal{G}\left(\binom{k}{k-p}\right) = \\ &= \mathcal{G}\left(\binom{-k+k-p+1}{k-p}(-1)^{k-p}\right) = \\ &= \mathcal{G}\left(\binom{-p-1}{k-p}(-1)^{k-p}\right) = \\ &= \mathcal{G}\left([t^{k-p}]\frac{1}{(1-t)^{p+1}}\right) = \\ &= \mathcal{G}\left([t^k]\frac{t^p}{(1-t)^{p+1}}\right) = \frac{t^p}{(1-t)^{p+1}}. \end{aligned}$$

Several generating functions for different forms of binomial coefficients can be found by means of this method. They are summarized as follows, where p and m are two parameters and can also be zero:

$$\begin{aligned} \mathcal{G}\left(\binom{p}{m+k}\right) &= \frac{(1+t)^p}{t^m} \\ \mathcal{G}\left(\binom{p+k}{m}\right) &= \frac{t^{m-p}}{(1-t)^{m+1}} \\ \mathcal{G}\left(\binom{p+k}{m+k}\right) &= \frac{1}{t^m(1-t)^{p+1-m}} \end{aligned}$$

These functions can make sense even when they are f.L.s. and not simply f.p.s..

Finally, we list the following generating functions:

$$\begin{aligned} \mathcal{G}\left(k\binom{p}{k}\right) &= tD\mathcal{G}\left(\binom{p}{k}\right) = \\ &= tD(1+t)^p = pt(1+t)^{p-1} \\ \mathcal{G}\left(k^2\binom{p}{k}\right) &= (tD+t^2D^2)\mathcal{G}\left(\binom{p}{k}\right) = \\ &= pt(1+pt)(1+t)^{p-2} \\ \mathcal{G}\left(\frac{1}{k+1}\binom{p}{k}\right) &= \frac{1}{t}\int_0^t \mathcal{G}\left(\binom{p}{k}\right)dz = \\ &= \frac{1}{t}\left[\frac{(1+t)^{p+1}}{p+1}\right]_0^t = \\ &= \frac{(1+t)^{p+1}-1}{(p+1)t} \end{aligned}$$

$$\begin{aligned}
\mathcal{G}\left(k\binom{k}{m}\right) &= tD\frac{t^m}{(1-t)^{m+1}} = \\
&= \frac{mt^m + t^{m+1}}{(1-t)^{m+2}} \\
\mathcal{G}\left(k^2\binom{k}{m}\right) &= (tD + t^2D^2)\mathcal{G}\left(\binom{k}{m}\right) = \\
&= \frac{m^2t^m + (3m+1)t^{m+1} + t^{m+2}}{(1-t)^{m+3}} \\
\mathcal{G}\left(\frac{1}{k}\binom{k}{m}\right) &= \int_0^t \left(\mathcal{G}\left(\binom{k}{m}\right) - \binom{0}{m}\right)\frac{dz}{z} = \\
&= \int_0^t \frac{z^{m-1}\,dz}{(1-z)^{m+1}} = \frac{t^m}{m(1-t)^m}
\end{aligned}$$

The last integral can be solved by setting $y = (1-z)^{-1}$ and is valid for $m > 0$; for $m = 0$ it reduces to $\mathcal{G}(1/k) = -\ln(1-t)$.

4.5 The Method of Shifting

When the elements of a sequence F are given by an explicit formula, we can try to find the generating function for F by using the technique of *shifting*: we consider the element f_{n+1} and try to express it in terms of f_n. This can produce a relation to which we apply the principle of identity deriving an equation in $\mathcal{G}(f_n)$, the solution of which is the generating function. In practice, we find a recurrence for the elements $f_n \in F$ and then try to solve it by using the rules $(G1) \div (G5)$ and their consequences. It can happen that the recurrence involves several elements in F and/or that the resulting equation is indeed a differential equation. Whatever the case, the method of shifting allows us to find the generating function of many sequences.

Let us consider the geometric sequence $(1, p, p^2, p^3, \ldots)$; we have $p^{k+1} = pp^k, \forall k \in \mathbb{N}$ or, by applying the operator $\mathcal{G}$, $\mathcal{G}(p^{k+1}) = p\mathcal{G}(p^k)$. By $(G2)$ we have $t^{-1}\left(\mathcal{G}(p^k) - 1\right) = p\mathcal{G}(p^k)$, that is:

$$\mathcal{G}(p^k) = \frac{1}{1-pt}$$

From this we obtain other generating functions:

$$\begin{aligned}
\mathcal{G}(kp^k) &= tD\frac{1}{1-pt} = \frac{pt}{(1-pt)^2} \\
\mathcal{G}(k^2p^k) &= tD\frac{pt}{(1-pt)^2} = \frac{pt + p^2t^2}{(1-pt)^3} \\
\mathcal{G}\left(\frac{1}{k}p^k\right) &= \int_0^t \left(\frac{1}{1-pz} - 1\right)\frac{dz}{z} = \\
&= \int \frac{p\,dz}{(1-pz)} = \ln\frac{1}{1-pt} \\
\mathcal{G}\left(\sum_{k=0}^n p^k\right) &= \frac{1}{1-t}\frac{1}{1-pt} = \\
&= \frac{1}{(p-1)t}\left(\frac{1}{1-pt} - \frac{1}{1-t}\right).
\end{aligned}$$

The last relation has been obtained by partial fraction expansion. By using the operator $[t^k]$ we easily find:

$$\begin{aligned}
\sum_{k=0}^n p^k &= [t^n]\frac{1}{1-t}\frac{1}{1-pt} = \\
&= \frac{1}{(p-1)}[t^{n+1}]\left(\frac{1}{1-pt} - \frac{1}{1-t}\right) = \\
&= \frac{p^{n+1}-1}{p-1}
\end{aligned}$$

the well-known formula for the sum of a geometric progression. We observe explicitly that the formulas above could have been obtained from formulas of the previous section and the general formula (4.2.10). In a similar way we also have:

$$\begin{aligned}
\mathcal{G}\left(\binom{m}{k}p^k\right) &= (1+pt)^m \\
\mathcal{G}\left(\binom{k}{m}p^k\right) &= \mathcal{G}\left(\binom{k}{k-m}p^{k-m}p^m\right) = \\
&= p^m\mathcal{G}\left(\binom{-m-1}{k-m}(-p)^{k-m}\right) = \\
&= \frac{p^mt^m}{(1-pt)^{m+1}}.
\end{aligned}$$

As a very simple application of the shifting method, let us observe that:

$$\frac{1}{(n+1)!} = \frac{1}{n+1}\frac{1}{n!}$$

that is $(n+1)f_{n+1} = f_n$, where $f_n = 1/n!$. By (4.2.4) we have $f'(t) = f(t)$ or:

$$\begin{aligned}
\mathcal{G}\left(\frac{1}{n!}\right) &= e^t \\
\mathcal{G}\left(\frac{1}{n\cdot n!}\right) &= \int_0^t \frac{e^z-1}{z}\,dz \\
\mathcal{G}\left(\frac{n}{(n+1)!}\right) &= tD\mathcal{G}\left(\frac{1}{(n+1)!}\right) = \\
&= tD\frac{1}{t}(e^t-1) = \frac{te^t - e^t + 1}{t}
\end{aligned}$$

By this relation the well-known result follows:

$$\begin{aligned}
\sum_{k=0}^n \frac{k}{(k+1)!} &= [t^n]\frac{1}{1-t}\left(\frac{te^t - e^t + 1}{t}\right) = \\
&= [t^n]\left(\frac{1}{1-t} - \frac{e^t-1}{t}\right) = \\
&= 1 - \frac{1}{(n+1)!}.
\end{aligned}$$

Let us now observe that $\binom{2n+2}{n+1} = \frac{2(2n+1)}{n+1}\binom{2n}{n}$; by setting $f_n = \binom{2n}{n}$, we have the recurrence $(n+$

$1)f_{n+1} = 2(2n+1)f_n$. By using (4.2.4), $(G1)$ and $(G3)$ we obtain the differential equation: $f'(t) = 4tf'(t) + 2f(t)$, the simple solution of which is:

$$\begin{aligned}
\mathcal{G}\left(\binom{2n}{n}\right) &= \frac{1}{\sqrt{1-4t}} \\
\mathcal{G}\left(\frac{1}{n+1}\binom{2n}{n}\right) &= \frac{1}{t}\int_0^t \frac{dz}{\sqrt{1-4z}} = \frac{1-\sqrt{1-4t}}{2t} \\
\mathcal{G}\left(\frac{1}{n}\binom{2n}{n}\right) &= \int_0^t \left(\frac{1}{\sqrt{1-4z}} - 1\right)\frac{dz}{z} = \\
&= 2\ln\frac{1-\sqrt{1-4t}}{2t} \\
\mathcal{G}\left(n\binom{2n}{n}\right) &= \frac{2t}{(1-4t)\sqrt{1-4t}} \\
\mathcal{G}\left(\frac{1}{2n+1}\binom{2n}{n}\right) &= \frac{1}{\sqrt{t}}\int_0^t \frac{dz}{\sqrt{4z(1-4z)}} = \\
&= \frac{1}{\sqrt{4t}}\arctan\sqrt{\frac{4t}{1-4t}}.
\end{aligned}$$

A last group of generating functions is obtained by considering $f_n = 4^n\binom{2n}{n}^{-1}$. Since:

$$4^{n+1}\binom{2n+2}{n+1}^{-1} = \frac{2n+2}{2n+1}4^n\binom{2n}{n}^{-1}$$

we have the recurrence: $(2n+1)f_{n+1} = 2(n+1)f_n$. By using the operator $\mathcal{G}$ and the rules of Section 4.2, the differential equation $2t(1-t)f'(t) - (1+2t)f(t) + 1 = 0$ is derived. The solution is:

$$f(t) = \sqrt{\frac{t}{(1-t)^3}}\left(-\int_0^t \sqrt{\frac{(1-z)^3}{z}}\frac{dz}{2z(1-z)}\right)$$

By simplifying and using the change of variable $y = \sqrt{z/(1-z)}$, the integral can be computed without difficulty, and the final result is:

$$\mathcal{G}\left(4^n\binom{2n}{n}^{-1}\right) = \sqrt{\frac{t}{(1-t)^3}}\arctan\sqrt{\frac{t}{1-t}} + \frac{1}{1-t}.$$

Some immediate consequences are:

$$\begin{aligned}
\mathcal{G}\left(\frac{4^n}{2n+1}\binom{2n}{n}^{-1}\right) &= \frac{1}{\sqrt{t(1-t)}}\arctan\sqrt{\frac{t}{1-t}} \\
\mathcal{G}\left(\frac{4^n}{2n^2}\binom{2n}{n}^{-1}\right) &= \left(\arctan\sqrt{\frac{t}{1-t}}\right)^2
\end{aligned}$$

and finally:

$$\mathcal{G}\left(\frac{1}{2n}\frac{4^n}{2n+1}\binom{2n}{n}^{-1}\right) = 1 - \sqrt{\frac{1-t}{t}}\arctan\sqrt{\frac{t}{1-t}}.$$

4.6 Diagonalization

The technique of shifting is a rather general method for obtaining generating functions. It produces first order recurrence relations, which will be more closely studied in the next sections. Not every sequence can be defined by a first order recurrence relation, and other methods are often necessary to find out generating functions. Sometimes, the rule of diagonalization can be used very conveniently. One of the most simple examples is how to determine the generating function of the central binomial coefficients, without having to pass through the solution of a differential equation. In fact we have:

$$\binom{2n}{n} = [t^n](1+t)^{2n}$$

and $(G6)$ can be applied with $F(t) = 1$ and $\phi(t) = (1+t)^2$. In this case, the function $w = w(t)$ is easily determined by solving the functional equation $w = t(1+w)^2$. By expanding, we find $tw^2 - (1-2t)w + t = 0$ or:

$$w = w(t) = \frac{1 - t \pm \sqrt{1-4t}}{2t}.$$

Since $w = w(t)$ should belong to $\mathcal{F}_1$, we must eliminate the solution with the + sign; consequently, we have:

$$\mathcal{G}\left(\binom{2n}{n}\right) =$$

$$= \left[\frac{1}{1-2t(1+w)} \;\middle|\; w = \frac{1-t-\sqrt{1-4t}}{2t}\right] =$$

$$= \frac{1}{\sqrt{1-4t}}$$

as we already know.

The function $\phi(t) = (1+t)^2$ gives rise to a second degree equation. More in general, let us study the sequence:

$$c_n = [t^n](1 + \alpha t + \beta t^2)^n$$

and look for its generating function $C(t) = \mathcal{G}(c_n)$. In this case again we have $F(t) = 1$ and $\phi(t) = 1 + \alpha t + \beta t^2$, and therefore we should solve the functional equation $w = t(1+\alpha w+\beta w^2)$ or $\beta t w^2 - (1-\alpha t)w + t = 0$. This gives:

$$w = w(t) = \frac{1 - \alpha t \pm \sqrt{(1-\alpha t)^2 - 4\beta t^2}}{2\beta t}$$

and again we have to eliminate the solution with the + sign. By performing the necessary computations, we find:

$$C(t) = \left[\frac{1}{1 - t(\alpha + 2\beta w)} \;\middle|\right.$$

$n\backslash k$	0	1	2	3	4	5	6	7	8
0	**1**								
1	1	**1**	1						
2	1	2	**3**	2	1				
3	1	3	6	**7**	6	3	1		
4	1	4	10	16	**19**	16	10	4	1

Table 4.2: Trinomial coefficients

$$\left| w = \frac{1-\alpha t-\sqrt{1-2\alpha t+(\alpha^2-4\beta)t^2}}{2\beta t} \right]$$

$$= \frac{1}{\sqrt{1-2\alpha t+(\alpha^2-4\beta)t^2}}$$

and for $\alpha = 2, \beta = 1$ we obtain again the generating function for the central binomial coefficients.

The coefficients of $(1+t+t^2)^n$ are called *trinomial coefficients*, in analogy with the binomial coefficients. They constitute an infinite array in which every row has two more elements, different from 0, with respect to the previous row (see Table 4.2).

If $T_{n,k}$ is $[t^k](1+t+t^2)^n$, a trinomial coefficient, by the obvious property $(1+t+t^2)^{n+1} = (1+t+t^2)(1+t+t^2)^n$, we immediately deduce the recurrence relation:

$$T_{n+1,k+1} = T_{n,k-1} + T_{n,k} + T_{n,k+1}$$

from which the array can be built, once we start from the initial conditions $T_{n,0} = 1$ and $T_{n,2n} = 1$, for every $n \in \mathbb{N}$. The elements $T_{n,n}$, marked in the table, are called the *central trinomial coefficients*; their sequence begins:

n	0	1	2	3	4	5	6	7	8
T_n	1	1	3	7	19	51	141	393	1107

and by the formula above their generating function is:

$$\mathcal{G}(T_{n,n}) = \frac{1}{\sqrt{1-2t-3t^2}} = \frac{1}{\sqrt{(1+t)(1-3t)}}.$$

4.7 Some special generating functions

We wish to determine the generating function of the sequence $\{0,1,1,1,2,2,2,2,3,\ldots\}$, that is the sequence whose generic element is $\lfloor\sqrt{k}\rfloor$. We can think that it is formed up by summing an infinite number of simpler sequences $\{0,1,1,1,1,1,1,\ldots\}$, $\{0,0,0,0,1,1,1,1,\ldots\}$, the next one with the first 1 in position 9, and so on. The generating functions of these sequences are:

$$\frac{t^1}{1-t} \qquad \frac{t^4}{1-t} \qquad \frac{t^9}{1-t} \qquad \frac{t^{16}}{1-t} \qquad \cdots$$

and therefore we obtain:

$$\mathcal{G}\Big(\lfloor\sqrt{k}\rfloor\Big) = \sum_{k=0}^{\infty} \frac{t^{k^2}}{1-t}.$$

In the same way we obtain analogous generating functions:

$$\mathcal{G}\Big(\lfloor\sqrt[r]{k}\rfloor\Big) = \sum_{k=0}^{\infty} \frac{t^{k^r}}{1-t} \quad \mathcal{G}(\lfloor\log_r k\rfloor) = \sum_{k=0}^{\infty} \frac{t^{r^k}}{1-t}$$

where r is any integer number, or also any real number, if we substitute to k^r and r^k, respectively, $\lceil k^r\rceil$ and $\lceil r^k\rceil$.

These generating functions can be used to find the values of several sums in closed or semi-closed form. Let us begin by the following case, where we use the Euler transformation:

$$\begin{aligned}
\sum_k \binom{n}{k}\lfloor\sqrt{k}\rfloor(-1)^k &= \\
&= (-1)^n \sum_k \binom{n}{k}(-1)^{n-k}\lfloor\sqrt{k}\rfloor = \\
&= (-1)^n[t^n]\frac{1}{1+t}\left[\sum_{k=0}^{\infty}\frac{y^{k^2}}{1-y}\;\Big|\; y=\frac{t}{1+t}\right] = \\
&= (-1)^n[t^n]\sum_{k=0}^{\infty}\frac{t^{k^2}}{(1+t)^{k^2}} = \\
&= (-1)^n\sum_{k=0}^{\lfloor\sqrt{n}\rfloor}[t^{n-k^2}]\frac{1}{(1+t)^{k^2}} = \\
&= (-1)^n\sum_{k=0}^{\lfloor\sqrt{n}\rfloor}\binom{-k^2}{n-k^2} = \\
&= (-1)^n\sum_{k=0}^{\lfloor\sqrt{n}\rfloor}\binom{n-1}{n-k^2}(-1)^{n-k^2} = \\
&= \sum_{k=0}^{\lfloor\sqrt{n}\rfloor}\binom{n-1}{n-k^2}(-1)^{k^2}.
\end{aligned}$$

We can think of the last sum as a "semi-closed" form, because the number of terms is dramatically reduced from n to $\sqrt{n}$, although it remains depending on n. In the same way we find:

$$\sum_k \binom{n}{k}\lfloor\log_2 k\rfloor(-1)^k = \sum_{k=1}^{\lfloor\log_2 n\rfloor}\binom{n-1}{n-2^k}.$$

A truly closed formula is found for the following sum:

$$\sum_{k=1}^{n}\lfloor\sqrt{k}\rfloor = [t^n]\frac{1}{1-t}\sum_{k=1}^{\infty}\frac{t^{k^2}}{1-t} =$$

$$
\begin{aligned}
&= \sum_{k=1}^{\infty}[t^{n-k^2}]\frac{1}{(1-t)^2} = \\
&= \sum_{k=1}^{\lfloor\sqrt{n}\rfloor}\binom{-2}{n-k^2}(-1)^{n-k^2} = \\
&= \sum_{k=1}^{\lfloor\sqrt{n}\rfloor}\binom{n-k^2+1}{n-k^2} = \\
&= \sum_{k=1}^{\lfloor\sqrt{n}\rfloor}(n-k^2+1) = \\
&= (n+1)\lfloor\sqrt{n}\rfloor - \sum_{k=1}^{\lfloor\sqrt{n}\rfloor}k^2.
\end{aligned}
$$

The final value of the sum is therefore:

$$(n+1)\lfloor\sqrt{n}\rfloor - \frac{\lfloor\sqrt{n}\rfloor}{3}\left(\lfloor\sqrt{n}\rfloor+1\right)\left(\lfloor\sqrt{n}\rfloor+\frac{1}{2}\right)$$

whose asymptotic value is $\frac{2}{3}n\sqrt{n}$. Again, for the analogous sum with $\lfloor\log_2 n\rfloor$, we obtain (see Chapter 1):

$$\sum_{k=1}^{n}\lfloor\log_2 k\rfloor = (n+1)\lfloor\log_2 n\rfloor - 2^{\lfloor\log_2 n\rfloor+1}+1.$$

A somewhat more difficult sum is the following one:

$$
\begin{aligned}
\sum_{k=0}^{n}\binom{n}{k}\lfloor\sqrt{k}\rfloor &= [t^n]\frac{1}{1-t}\left[\sum_{k=0}^{\infty}\frac{y^{k^2}}{1-y}\;\middle|\; y=\frac{t}{1-t}\right] \\
&= [t^n]\sum_{k=0}^{\infty}\frac{t^{k^2}}{(1-t)^{k^2}(1-2t)} = \\
&= \sum_{k=0}^{\infty}[t^{n-k^2}]\frac{1}{(1-t)^{k^2}(1-2t)}.
\end{aligned}
$$

We can now obtain a semi-closed form for this sum by expanding the generating function into partial fractions:

$$\frac{1}{(1-t)^{k^2}(1-2t)} = \frac{A}{1-2t}+\frac{B}{(1-t)^{k^2}}+$$
$$+\frac{C}{(1-t)^{k^2-1}}+\frac{D}{(1-t)^{k^2-2}}+\cdots+\frac{X}{1-t}.$$

We can show that $A=2^{k^2}, B=-1, C=-2, D=-4, \ldots, X=-2^{k^2-1}$; in fact, by substituting these values in the previous expression we get:

$$
\begin{aligned}
&\frac{2^{k^2}}{1-2t}-\frac{1}{(1-t)^{k^2}}-\frac{2(1-t)}{(1-t)^{k^2}}- \\
&\quad -\frac{4(1-t)^2}{(1-t)^{k^2}}-\cdots-\frac{2^{k^2-1}(1-t)^{k^2-1}}{(1-t)^{k^2}} = \\
&= \frac{2^{k^2}}{1-2t}-\frac{1}{(1-t)^{k^2}}\left(\frac{2^{k^2}(1-t)^{k^2}-1}{2(1-t)-1}\right) = \\
&= \frac{2^{k^2}}{1-2t}-\frac{2^{k^2}(1-t)^{k^2}-1}{(1-t)^{k^2}(1-2t)} = \\
&= \frac{1}{(1-t)^{k^2}(1-2t)}.
\end{aligned}
$$

Therefore, we conclude:

$$\sum_{k=0}^{n}\binom{n}{k}\lfloor\sqrt{k}\rfloor = \sum_{k=1}^{\lfloor\sqrt{n}\rfloor}2^{k^2}2^{n-k^2}-\sum_{k=1}^{\lfloor\sqrt{n}\rfloor}\binom{n-1}{n-k^2}-$$
$$-\sum_{k=1}^{\lfloor\sqrt{n}\rfloor}2\binom{n-2}{n-k^2}-\cdots-\sum_{k=1}^{\lfloor\sqrt{n}\rfloor}2^{k^2-1}\binom{n-k^2}{n-k^2} =$$
$$= \lfloor\sqrt{n}\rfloor 2^n - \sum_{k=1}^{\lfloor\sqrt{n}\rfloor}\left(\binom{n-1}{n-k^2}+2\binom{n-2}{n-k^2}+\cdots+\right.$$
$$\left.+\cdots+2^{k^2-1}\binom{n-k^2}{n-k^2}\right).$$

We observe that for very large values of n the first term dominates all the others, and therefore the asymptotic value of the sum is $\lfloor\sqrt{n}\rfloor 2^n$.

4.8 Linear recurrences with constant coefficients

If $(f_k)_{k\in N}$ is a sequence, it can be defined by means of a *recurrence relation*, i.e., a relation relating the generic element f_n to other elements f_k having $k<n$. Usually, the first elements of the sequence must be given explicitly, in order to allow the computation of the successive values; they constitute the *initial conditions* and the sequence is well-defined if and only if every element can be computed by starting with the initial conditions and going on with the other elements by means of the recurrence relation. For example, the constant sequence $(1,1,1,\ldots)$ can be defined by the recurrence relation $x_n = x_{n-1}$ and the initial condition $x_0=1$. By changing the initial conditions, the sequence can radically change; if we consider the same relation $x_n = x_{n-1}$ together with the initial condition $x_0=2$, we obtain the constant sequence $\{2,2,2,\ldots\}$.

In general, a recurrence relation can be written $f_n = F(f_{n-1}, f_{n-2}, \ldots)$; when F depends on all the values $f_{n-1}, f_{n-2}, \ldots, f_1, f_0$, then the relation is called a *full history recurrence*. If F only depends on a fixed number of elements $f_{n-1}, f_{n-2}, \ldots, f_{n-p}$, then the relation is called a *partial history recurrence* and p is called the *order* of the relation. Besides, if F is linear, we have a *linear recurrence*. Linear recurrences are surely the most common and important type of recurrence relations; if all the coefficients appearing in F are constant, we have a linear recurrence *with*

constant coefficients, and if the coefficients are polynomials in n, we have a linear recurrence *with polynomial coefficients*. As we are now going to see, the method of generating functions allows us to find the solution of any linear recurrence with constant coefficients, in the sense that we find a function $f(t)$ such that $[t^n]f(t) = f_n, \forall n \in \mathbb{N}$. For linear recurrences with polynomial coefficients, the same method allows us to find a solution in many occasions, but the success is not assured. On the other hand, no method is known that solves all the recurrences of this kind, and surely generating functions are the method giving the highest number of positive results. We will discuss this case in the next section.

The Fibonacci recurrence $F_n = F_{n-1} + F_{n-2}$ is an example of a recurrence relation with constant coefficients. When we have a recurrence of this kind, we begin by expressing it in such a way that the relation is valid for every $n \in \mathbb{N}$. In the example of Fibonacci numbers, this is not the case, because for $n = 0$ we have $F_0 = F_{-1} + F_{-2}$, and we do not know the values for the two elements in the r.h.s., which have no combinatorial meaning. However, if we write the recurrence as $F_{n+2} = F_{n+1} + F_n$ we have fulfilled the requirement. This first step has a great importance, because it allows us to apply the operator $\mathcal{G}$ to both members of the relation; this was not possible beforehand because of the principle of identity for generating functions.

The recurrence being linear with constant coefficients, we can apply the axiom of linearity to the recurrence:

$$f_{n+p} = \alpha_1 f_{n+p-1} + \alpha_2 f_{n+p-2} + \cdots + \alpha_p f_n$$

and obtain the relation:

$$\mathcal{G}(f_{n+p}) = \alpha_1 \mathcal{G}(f_{n+p-1}) + \\ + \alpha_2 \mathcal{G}(f_{n+p-2}) + \cdots + \alpha_p \mathcal{G}(f_n).$$

By Theorem 4.2.2 we can now express every $\mathcal{G}(f_{n+p-j})$ in terms of $f(t) = \mathcal{G}(f_n)$ and obtain a linear relation in $f(t)$, from which an explicit expression for $f(t)$ is immediately obtained. This is the solution of the recurrence relation. We observe explicitly that in writing the expressions for $\mathcal{G}(f_{n+p-j})$ we make use of the initial conditions for the sequence.

Let us go on with the example of the Fibonacci sequence $(F_k)_{k \in N}$. We have:

$$\mathcal{G}(F_{n+2}) = \mathcal{G}(F_{n+1}) + \mathcal{G}(F_n)$$

and by setting $F(t) = \mathcal{G}(F_n)$ we find:

$$\frac{F(t) - F_0 - F_1 t}{t^2} = \frac{F(t) - F_0}{t} + F(t).$$

Because we know that $F_0 = 0, F_1 = 1$, we have:

$$F(t) - t = tF(t) + t^2 F(t)$$

and by solving in $F(t)$ we have the explicit expression:

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

This is the generating function for the Fibonacci numbers. We can now find an explicit expression for F_n in the following way. The denominator of $F(t)$ can be written $1 - t - t^2 = (1 - \phi t)(1 - \widehat{\phi} t)$ where:

$$\phi = \frac{1 + \sqrt{5}}{2} \approx 1.618033989$$

$$\widehat{\phi} = \frac{1 - \sqrt{5}}{2} \approx -0.618033989.$$

The constant $1/\phi \approx 0.618033989$ is known as the *golden ratio*. By applying the method of *partial fraction expansion* we find:

$$\begin{aligned} F(t) &= \frac{t}{(1 - \phi t)(1 - \widehat{\phi} t)} = \frac{A}{1 - \phi t} + \frac{B}{1 - \widehat{\phi} t} = \\ &= \frac{A - A\widehat{\phi} t + B - B\phi t}{1 - t - t^2}. \end{aligned}$$

We determine the two constants A and B by equating the coefficients in the first and last expression for $F(t)$:

$$\begin{cases} A + B = 0 \\ -A\widehat{\phi} - B\phi = 1 \end{cases} \qquad \begin{cases} A = 1/(\phi - \widehat{\phi}) = 1/\sqrt{5} \\ B = -A = -1/\sqrt{5} \end{cases}$$

The value of F_n is now obtained by extracting the coefficient of t^n:

$$\begin{aligned} F_n &= [t^n]F(t) = \\ &= [t^n]\frac{1}{\sqrt{5}}\left(\frac{1}{1 - \phi t} - \frac{1}{1 - \widehat{\phi} t}\right) = \\ &= \frac{1}{\sqrt{5}}\left([t^n]\frac{1}{1 - \phi t} - [t^n]\frac{1}{1 - \widehat{\phi} t}\right) = \\ &= \frac{\phi^n - \widehat{\phi}^n}{\sqrt{5}}. \end{aligned}$$

This formula allows us to compute F_n in a time independent of n, because $\phi^n = \exp(n \ln \phi)$, and shows that F_n grows exponentially. In fact, since $|\widehat{\phi}| < 1$, the quantity $\widehat{\phi}^n$ approaches 0 very rapidly and we have $F_n = O(\phi^n)$. In reality, F_n should be an integer and therefore we can compute it by finding the integer number closest to $\phi^n/\sqrt{5}$; consequently:

$$F_n = \text{round}\left(\frac{\phi^n}{\sqrt{5}}\right).$$

4.9 Linear recurrences with polynomial coefficients

When a recurrence relation has polynomial coefficients, the method of generating functions does not assure a solution, but no other method is available to solve those recurrences which cannot be solved by a generating function approach. Usually, the rule of differentiation introduces a derivative in the relation for the generating function and a differential equation has to be solved. This is the actual problem of this approach, because the main difficulty just consists in dealing with the differential equation. We have already seen some examples when we studied the method of shifting, but here we wish to present a case arising from an actual combinatorial problem, and in the next section we will see a very important example taken from the analysis of algorithms.

When we studied permutations, we introduced the concept of an *involution*, i.e., a permutation $\pi \in P_n$ such that $\pi^2 = (1)$, and for the number I_n of involutions in P_n we found the recurrence relation:

$$I_n = I_{n-1} + (n-1)I_{n-2}$$

which has polynomial coefficients. The number of involutions grows very fast and it can be a good idea to consider the quantity $i_n = I_n/n!$. Therefore, let us begin by changing the recurrence in such a way that the principle of identity can be applied, and then divide everything by $(n+2)!$:

$$\begin{aligned} I_{n+2} &= I_{n+1} + (n+1)I_n \\ \frac{I_{n+2}}{(n+2)!} &= \frac{1}{n+2}\frac{I_{n+1}}{(n+1)!} + \frac{1}{n+2}\frac{I_n}{n!}. \end{aligned}$$

The recurrence relation for i_n is:

$$(n+2)i_{n+2} = i_{n+1} + i_n$$

and we can pass to generating functions. $\mathcal{G}((n+2)i_{n+2})$ can be seen as the shifting of $\mathcal{G}((n+1)i_{n+1}) = i'(t)$, if with $i(t)$ we denote the generating function of $(i_k)_{k\in N} = (I_k/k!)_{k\in N}$, which is therefore the exponential generating function for $(I_k)_{k\in N}$. We have:

$$\frac{i'(t)-1}{t} = \frac{i(t)-1}{t} + i(t)$$

because of the initial conditions $i_0 = i_1 = 1$, and so:

$$i'(t) = (1+t)i(t).$$

This is a simple differential equation with separable variables and by solving it we find:

$$\ln i(t) = t + \frac{t^2}{2} + C \quad \text{or} \quad i(t) = \exp\left(t + \frac{t^2}{2} + C\right)$$

where C is an integration constant. Because $i(0) = e^C = 1$, we have $C = 0$ and we conclude with the formula:

$$I_n = n![t^n]\exp\left(t + \frac{t^2}{2}\right).$$

4.10 The summing factor method

For linear recurrences of the first order a method exists, which allows us to obtain an explicit expression for the generic element of the defined sequence. Usually, this expression is in the form of a sum, and a possible closed form can only be found by manipulating this sum; therefore, the method does not guarantee a closed form. Let us suppose we have a recurrence relation:

$$a_{n+1}f_{n+1} = b_n f_n + c_n$$

where a_n, b_n, c_n are any expressions, possibly depending on n. As we remarked in the Introduction, if $a_{n+1} = b_n = 1$, by *unfolding* the recurrence we can find an explicit expression for f_{n+1} or f_n:

$$f_{n+1} = f_n + c_n = f_{n-1} + c_{n-1} + c_n = \cdots = f_0 + \sum_{k=0}^{n} c_k$$

where f_0 is the initial condition relative to the sequence under consideration. Fortunately, we can always change the original recurrence into a relation of this more simple form. In fact, if we multiply everything by the so-called *summing factor*:

$$\frac{a_n a_{n-1}\dots a_0}{b_n b_{n-1}\dots b_0}$$

provided none of $a_n, a_{n-1}, \dots, a_0, b_n, b_{n-1}, \dots, b_0$ is zero, we obtain:

$$\begin{aligned} &\frac{a_{n+1}a_n\dots a_0}{b_n b_{n-1}\dots b_0}f_{n+1} = \\ &\quad = \frac{a_n a_{n-1}\dots a_0}{b_{n-1}b_{n-2}\dots b_0}f_n + \frac{a_n a_{n-1}\dots a_0}{b_n b_{n-1}\dots b_0}c_n. \end{aligned}$$

We can now define:

$$g_{n+1} = a_{n+1}a_n\dots a_0 f_{n+1}/(b_n b_{n-1}\dots b_0),$$

and the relation becomes:

$$g_{n+1} = g_n + \frac{a_n a_{n-1}\dots a_0}{b_n b_{n-1}\dots b_0}c_n \qquad g_0 = a_0 f_0.$$

Finally, by unfolding this recurrence the result is:

$$f_{n+1} = \frac{b_n b_{n-1}\dots b_0}{a_{n+1}a_n\dots a_0}\left(a_0 f_0 + \sum_{k=0}^{n}\frac{a_k a_{k-1}\dots a_0}{b_k b_{k-1}\dots b_0}c_k\right).$$

As a technical remark, we observe that sometimes a_0 and/or b_0 can be 0; in that case, we can unfold the

recurrence down to 1, and accordingly change the last index 0.

In order to show a non-trivial example, let us discuss the problem of determining the coefficient of t^n in the f.p.s. corresponding to the function $f(t) = \sqrt{1-t}\ln(1/(1-t))$. If we expand the function, we find:

$$\begin{aligned} f(t) &= \sqrt{1-t}\ln\frac{1}{1-t} = \\ &= t - \frac{1}{24}t^3 - \frac{1}{24}t^4 - \frac{71}{1920}t^5 - \frac{31}{960}t^6 + \cdots. \end{aligned}$$

A method for finding a recurrence relation for the coefficients f_n of this f.p.s. is to derive a differential equation for $f(t)$. By differentiating:

$$f'(t) = -\frac{1}{2\sqrt{1-t}}\ln\frac{1}{1-t} + \frac{1}{\sqrt{1-t}}$$

and therefore we have the differential equation:

$$(1-t)f'(t) = -\frac{1}{2}f(t) + \sqrt{1-t}.$$

By extracting the coefficient of t^n, we have the relation:

$$(n+1)f_{n+1} - nf_n = -\frac{1}{2}f_n + \binom{1/2}{n}(-1)^n$$

which can be written as:

$$(n+1)f_{n+1} = \frac{2n-1}{2}f_n - \frac{1}{4^n(2n-1)}\binom{2n}{n}.$$

This is a recurrence relation of the first order with the initial condition $f_0 = 0$. Let us apply the summing factor method, for which we have $a_n = n$, $b_n = (2n-1)/2$. Since $a_0 = 0$, we have:

$$\begin{aligned} \frac{a_n a_{n-1}\ldots a_1}{b_n b_{n-1}\ldots b_1} &= \frac{n(n-1)\cdots 1\cdot 2^n}{(2n-1)(2n-3)\cdots 1} = \\ &= \frac{n!2^n 2n(2n-2)\cdots 2}{2n(2n-1)(2n-2)\cdots 1} = \\ &= \frac{4^n n!^2}{(2n)!}. \end{aligned}$$

By multiplying the recurrence relation by this summing factor, we find:

$$(n+1)\frac{4^n n!^2}{(2n)!}f_{n+1} = \frac{2n-1}{2}\frac{4^n n!^2}{(2n)!}f_n - \frac{1}{2n-1}.$$

We are fortunate and c_n simplifies dramatically; besides, we know that the two coefficients of f_{n+1} and f_n are equal, notwithstanding their appearance. Therefore we have:

$$f_{n+1} = \frac{1}{(n+1)4^n}\binom{2n}{n}\sum_{k=0}^{n}\frac{-1}{2k-1} \quad (a_0 f_0 = 0).$$

We can somewhat simplify this expression by observing that:

$$\sum_{k=0}^{n}\frac{1}{2k-1} = \frac{1}{2n-1} + \frac{1}{2n-3} + \cdots + 1 - 1 =$$

$$= \frac{1}{2n} + \frac{1}{2n-1} + \frac{1}{2n-2} + \cdots + \frac{1}{2} + 1 - \frac{1}{2n} - \cdots - \frac{1}{2} - 1 =$$

$$= H_{2n+2} - \frac{1}{2n+1} - \frac{1}{2n+2} - \frac{1}{2}H_{n+1} + \frac{1}{2n+2} - 1 =$$

$$= H_{2n+2} - \frac{1}{2}H_{n+1} - \frac{2(n+1)}{2n+1}.$$

Furthermore, we have:

$$\begin{aligned} \frac{1}{(n+1)4^n}\binom{2n}{n} &= \frac{4}{(n+1)4^{n+1}}\binom{2n+2}{n+1}\frac{n+1}{2(2n+1)} \\ &= \frac{2}{2n+1}\frac{1}{4^{n+1}}\binom{2n+2}{n+1}. \end{aligned}$$

Therefore:

$$f_{n+1} = \left(\frac{1}{2}H_{n+1} - H_{2n+2} + \frac{2(n+1)}{2n+1}\right) \times$$

$$\times\frac{2}{2n+1}\frac{1}{4^{n+1}}\binom{2n+2}{n+1}.$$

This expression allows us to obtain a formula for f_n:

$$f_n = \left(\frac{1}{2}H_n - H_{2n} + \frac{2n}{2n-1}\right)\frac{2}{2n-1}\frac{1}{4^n}\binom{2n}{n} =$$

$$= \left(H_n - 2H_{2n} + \frac{4n}{2n-1}\right)\binom{1/2}{n}.$$

The reader can numerically check this expression against the actual values of f_n given above. By using the asymptotic approximation $H_n \sim \ln n + \gamma$ given in the Introduction, we find:

$$\begin{aligned} H_n - 2H_{2n} + \frac{4n}{2n-1} &\sim \\ &\sim \ln n + \gamma - 2(\ln 2 + \ln n + \gamma) + 2 = \\ &= -\ln n - \gamma - \ln 4 + 2. \end{aligned}$$

Besides:

$$\frac{1}{2n-1}\frac{1}{4^n}\binom{2n}{n} \sim \frac{1}{2n\sqrt{\pi n}}$$

and we conclude:

$$f_n \sim -(\ln n + \gamma + \ln 4 - 2)\frac{1}{2n\sqrt{\pi n}}$$

which shows that $|f_n|$ grows as $\ln n/n^{3/2}$.

4.11 The internal path length of binary trees

Binary trees are often used as a data structure to retrieve information. A set D of keys is given, taken from an ordered universe U. Therefore D is a permutation of the ordered sequence $d_1 < d_2 < \cdots < d_n$, and as the various elements arrive, they are inserted in a binary tree. As we know, there are $n!$ possible permutations of the keys in D, but there are only $\binom{2n}{n}/(n+1)$ different binary trees with n nodes. When we are looking for some key d to find whether $d \in D$ or not, we perform a search in the binary tree, comparing d against the root and other keys in the tree, until we find d or we arrive to some leaf and we cannot go on with our search. In the former case our search is successful, while in the latter case it is unsuccessful. The problem is: how many comparisons should we perform, on the average, to find out that d is present in the tree (successful search)? The answer to this question is very important, because it tells us how good binary trees are for searching information.

The number of nodes along a path in the tree, starting at the root and arriving to a given node K is called the *internal path length* for K. It is just the number of comparisons necessary to find the key in K. Therefore, our previous problem can be stated in the following way: what is the average internal path length for binary trees with n nodes? Knuth has found a rather simple way for answering this question; however, we wish to show how the method of generating functions can be used to find the average internal path length in a standard way. The reasoning is as follows: we evaluate the total internal path length for all the trees generated by the $n!$ possible permutations of our key set D, and then divide this number by $n!n$, the total number of nodes in all the trees.

A non-empty binary tree can be seen as two subtrees connected to the root (see Section 2.9); the left subtree contains k nodes ($k = 0, 1, \ldots, n-1$) and the right subtree contains the remaining $n-1-k$ nodes. Let P_n the total internal path length (i.p.l.) of all the $n!$ possible trees generated by permutations. The left subtrees have therefore a total i.p.l. equal to P_k, but every search in these subtrees has to pass through the root. This increases the total i.p.l. by the total number of nodes, i.e., it actually is $P_k + k!k$. We now observe that every left subtree is associated to each possible right subtree and therefore it should be counted $(n-1-k)!$ times. Besides, every permutation generating the left and right subtrees is not to be counted only once: the keys can be arranged in all possible ways in the overall permutation, retaining their relative ordering. These possible ways are $\binom{n-1}{k}$, and therefore the total contribution to P_n of the left subtrees is:

$$\binom{n-1}{k}(n-1-k)!(P_k + k!k) = (n-1)!\left(\frac{P_k}{k!} + k\right).$$

In a similar way we find the total contribution of the right subtrees:

$$\binom{n-1}{k}k!(P_{n-1-k} + (n-1-k)!(n-1-k)) =$$

$$= (n-1)!\left(\frac{P_{n-1-k}}{(n-1-k)!} + (n-1-k)\right).$$

It only remains to count the contribution of the roots, which obviously amounts to $n!$, a single comparisons for every tree. We therefore have the following recurrence relation, in which the contributions of the left and right subtrees turn out to be the same:

$$P_n = n! + (n-1)!\times$$

$$\times \sum_{k=0}^{n-1}\left(\frac{P_k}{k!} + k + \frac{P_{n-1-k}}{(n-1-k)!} + (n-1-k)\right) =$$

$$\begin{aligned} &= n! + 2(n-1)!\sum_{k=0}^{n-1}\left(\frac{P_k}{k!} + k\right) = \\ &= n! + 2(n-1)!\left(\sum_{k=0}^{n-1}\frac{P_k}{k!} + \frac{n(n-1)}{2}\right). \end{aligned}$$

We used the formula for the sum of the first $n-1$ integers, and now, by dividing by $n!$, we have:

$$\frac{P_n}{n!} = 1 + \frac{2}{n}\sum_{k=0}^{n-1}\frac{P_k}{k!} + n - 1 = n + \frac{2}{n}\sum_{k=0}^{n-1}\frac{P_k}{k!}.$$

Let us now set $Q_n = P_n/n!$, so that Q_n is the average total i.p.l. relative to a single tree. If we'll succeed in finding Q_n, the average i.p.l. we are looking for will simply be Q_n/n. We can also reformulate the recurrence for $n+1$, in order to be able to apply the generating function operator:

$$\begin{aligned} (n+1)Q_{n+1} &= (n+1)^2 + 2\sum_{k=0}^{n} Q_k \\ Q'(t) &= \frac{1+t}{(1-t)^3} + 2\frac{Q(t)}{1-t} \\ Q'(t) &- \frac{2}{1-t}Q(t) = \frac{1+t}{(1-t)^3}. \end{aligned}$$

This differential equation can be easily solved:

$$\begin{aligned} Q(t) &= \frac{1}{(1-t)^2}\left(\int \frac{(1-t)^2(1+t)}{(1-t)^3}dt + C\right) = \\ &= \frac{1}{(1-t)^2}\left(2\ln\frac{1}{1-t} - t + C\right). \end{aligned}$$

Since the i.p.l. of the empty tree is 0, we should have $Q_0 = Q(0) = 0$ and therefore, by setting $t = 0$, we find $C = 0$. The final result is:

$$Q(t) = \frac{2}{(1-t)^2} \ln \frac{1}{1-t} - \frac{t}{(1-t)^2}.$$

We can now use the formula for $\mathcal{G}(nH_n)$ (see the Section 4.4 on Common Generating Functions) to extract the coefficient of t^n:

$$Q_n = [t^n] \frac{1}{(1-t)^2} \left(2 \left(\ln \frac{1}{1-t} + 1 \right) - (2+t) \right) =$$

$$= 2[t^{n+1}] \frac{t}{(1-t)^2} \left(\ln \frac{1}{1-t} + 1 \right) - [t^n] \frac{2+t}{(1-t)^2} =$$

$$= 2(n+1)H_{n+1} - 2\binom{-2}{n}(-1)^n - \binom{-2}{n-1}(-1)^{n-1} =$$

$$= 2(n+1)H_n + 2 - 2(n+1) - n = 2(n+1)H_n - 3n.$$

Thus we conclude with the average i.p.l.:

$$\frac{P_n}{n!n} = \frac{Q_n}{n} = 2 \left(1 + \frac{1}{n} \right) H_n - 3.$$

This formula is asymptotic to $2 \ln n + \gamma - 3$, and shows that the average number of comparisons necessary to retrieve any key in a binary tree is in the order of $O(\ln n)$.

4.12 Height balanced binary trees

We have been able to show that binary trees are a "good" retrieving structure, in the sense that if the elements, or keys, of a set $\{a_1, a_2, \ldots, a_n\}$ are stored in random order in a binary (search) tree, then the expected average time for retrieving any key in the tree is in the order of $\ln n$. However, this behavior of binary trees is not always assured; for example, if the keys are stored in the tree in their proper order, the resulting structure degenerates into a linear list and the average retrieving time becomes $O(n)$.

To avoid this drawback, at the beginning of the 1960's, two Russian researchers, Adelson-Velski and Landis, found an algorithm to store keys in a "height balanced" binary tree, a tree for which the height of the left subtree of every node K differs by at most 1 from the height of the right subtree of the same node K. To understand this concept, let us define the *height* of a tree as the highest level at which a node in the tree is placed. The height is also the maximal number of comparisons necessary to find any key in the tree. Therefore, if we find a limitation for the height of a class of trees, this is also a limitation for the internal path length of the trees in the same class. Formally, a *height balanced binary tree* is a tree such that for every node K in it, if h'_K and h''_k are the heights of the two subtrees originating from K, then $|h'_K - h''_K| \leq 1$.

The algorithm of Adelson-Velski and Landis is very important because, as we are now going to show, height balanced binary trees assure that the retrieving time for every key in the tree is $O(\ln n)$. Because of that, height balanced binary trees are also known as *AVL trees*, and the algorithm for building AVL-trees from a set of n keys can be found in any book on algorithms and data structures. Here we only wish to perform a worst case analysis to prove that the retrieval time in any AVL tree cannot be larger than $O(\ln n)$.

In order to perform our analysis, let us consider to worst possible AVL tree. Since, by definition, the height of the left subtree of any node cannot exceed the height of the corresponding right subtree plus 1, let us consider trees in which the height of the left subtree of every node exceeds exactly by 1 the height of the right subtree of the same node. In Figure 4.1 we have drawn the first cases. These trees are built in a very simple way: every tree T_n, of height n, is built by using the preceding tree T_{n-1} as the left subtree and the tree T_{n-2} as the right subtree of the root. Therefore, the number of nodes in T_n is just the sum of the nodes in T_{n-1} and in T_{n-2}, plus 1 (the root), and the condition on the heights of the subtrees of every node is satisfied. Because of this construction, T_n can be considered as the "worst" tree of height n, in the sense that every other AVL-tree of height n will have at least as many nodes as T_n. Since the height n is an upper bound for the number of comparisons necessary to retrieve any key in the tree, the average retrieving time for every such tree will be $\leq n$.

If we denote by $|T_n|$ the number of nodes in the tree T_n, we have the simple recurrence relation:

$$|T_n| = |T_{n-1}| + |T_{n-2}| + 1.$$

This resembles the Fibonacci recurrence relation, and, in fact, we can easily show that $|T_n| = F_{n+1} - 1$, as is intuitively apparent from the beginning of the sequence $\{0, 1, 2, 4, 7, 12, \ldots\}$. The proof is done by mathematical induction. For $n = 0$ we have $|T_0| = F_1 - 1 = 1 - 1 = 0$, and this is true; similarly we proceed for $n + 1$. Therefore, let us suppose that for every $k < n$ we have $|T_{k+1}| = F_k - 1$; this holds for $k = n-1$ and $k = n-2$, and because of the recurrence relation for $|T_n|$ we have:

$$\begin{aligned} |T_n| &= |T_{n-1}| + |T_{n-2}| + 1 = \\ &= F_n - 1 + F_{n-1} - 1 + 1 = F_{n+1} - 1 \end{aligned}$$

since $F_n + F_{n-1} = F_{n+1}$ by the Fibonacci recurrence.

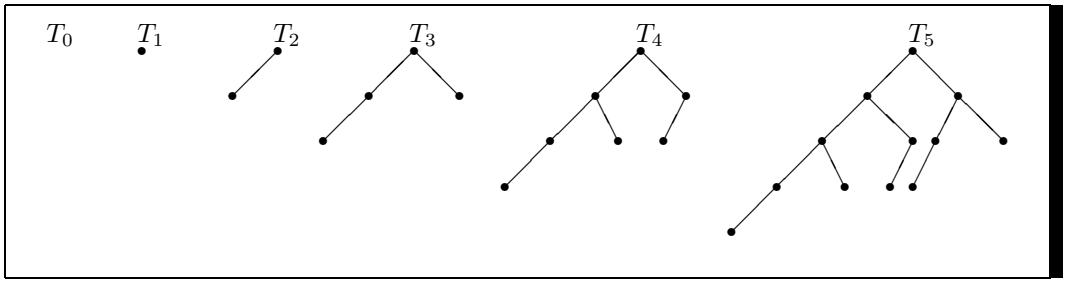


Figure 4.1: Worst AVL-trees

We have shown that for large values of n we have $F_n \approx \phi^n/\sqrt{5}$; therefore we have $|T_n| \approx \phi^{n+1}/\sqrt{5} - 1$ or $\phi^{n+1} \approx \sqrt{5}(|T_n|+1)$. By passing to the logarithms, we have: $n \approx \log_\phi(\sqrt{5}(|T_n|+1)) - 1$, and since all logarithms are proportional, $n = O(\ln |T_n|)$. As we observed, every AVL-tree of height n has a number of nodes not less than $|T_n|$, and this assures that the retrieving time for every AVL-tree with at most $|T_n|$ nodes is bounded from above by $\log_\phi(\sqrt{5}(|T_n|+1))-1$

4.13 Some special recurrences

Not all recurrence relations are linear and we had occasions to deal with a different sort of relation when we studied the Catalan numbers. They satisfy the recurrence $C_n = \sum_{k=0}^{n-1} C_k C_{n-k-1}$, which however, in this particular form, is only valid for $n > 0$. In order to apply the method of generating functions, we write it for $n+1$:

$$C_{n+1} = \sum_{k=0}^{n} C_k C_{n-k}.$$

The right hand member is a convolution, and therefore, by the initial condition $C_0 = 1$, we obtain:

$$\frac{C(t)-1}{t} = C(t)^2 \quad \text{or} \quad tC(t)^2 - C(t) + 1 = 0.$$

This is a second degree equation, which can be directly solved; for $t = 0$ we should have $C(0) = C_0 = 1$, and therefore the solution with the + sign before the square root is to be ignored; we thus obtain:

$$C(t) = \frac{1 - \sqrt{1-4t}}{2t}$$

which was found in the section "The method of shifting" in a completely different way.

The Bernoulli numbers were introduced by means of the implicit relation:

$$\sum_{k=0}^{n} \binom{n+1}{k} B_k = \delta_{n,0}.$$

We are now in a position to find out their exponential generating function, i.e., the function $B(t) = \mathcal{G}(B_n/n!)$, and prove some of their properties. The defining relation can be written as:

$$\begin{aligned}\sum_{k=0}^{n} \binom{n+1}{n-k} B_{n-k} &= \\ &= \sum_{k=0}^{n} (n+1)n\cdots(k+2)\frac{B_{n-k}}{(n-k)!} = \\ &= \sum_{k=0}^{n} \frac{(n+1)!}{(k+1)!}\frac{B_{n-k}}{(n-k)!} = \delta_{n,0}.\end{aligned}$$

If we divide everything by $(n+1)!$, we obtain:

$$\sum_{k=0}^{n} \frac{1}{(k+1)!}\frac{B_{n-k}}{(n-k)!} = \delta_{n,0}$$

and since this relation holds for every $n \in \mathbb{N}$, we can pass to the generating functions. The left hand member is a convolution, whose first factor is the shift of the exponential function and therefore we obtain:

$$\frac{e^t - 1}{t} B(t) = 1 \quad \text{or} \quad B(t) = \frac{t}{e^t - 1}.$$

The classical way to see that $B_{2n+1} = 0, \forall n > 0$, is to show that the function obtained from $B(t)$ by deleting the term of first degree is an even function, and therefore should have all its coefficients of odd order equal to zero. In fact we have:

$$B(t) + \frac{t}{2} = \frac{t}{e^t - 1} + \frac{t}{2} = \frac{t}{2}\frac{e^t + 1}{e^t - 1}.$$

In order to see that this function is even, we substitute $t \to -t$ and show that the function remains the same:

$$-\frac{t}{2}\frac{e^{-t}+1}{e^{-t}-1} = -\frac{t}{2}\frac{1+e^t}{1-e^t} = \frac{t}{2}\frac{e^t+1}{e^t-1}.$$

Chapter 5

Riordan Arrays

5.1 Definitions and basic concepts

A *Riordan array* is a couple of formal power series $D = (d(t), h(t))$; if both $d(t), h(t) \in \mathcal{F}_0$, then the Riordan array is called *proper*. The Riordan array can be identified with the infinite, lower triangular array (or *triangle*) $(d_{n,k})_{n,k\in N}$ defined by:

$$d_{n,k} = [t^n]d(t)(th(t))^k \tag{5.1.1}$$

In fact, we are mainly interested in the sequence of functions iteratively defined by:

$$\begin{cases} d_0(t) &= d(t) \\ d_k(t) &= d_{k-1}(t)th(t) = d(t)(th(t))^k \end{cases}$$

These functions are the column generating functions of the triangle.

Another way of characterizing a Riordan array $D = (d(t), h(t))$ is to consider the bivariate generating function:

$$d(t,z) = \sum_{k=0}^{\infty} d(t)(th(t))^k z^k = \frac{d(t)}{1-tzh(t)} \tag{5.1.2}$$

A common example of a Riordan array is the Pascal triangle, for which we have $d(t) = h(t) = 1/(1-t)$. In fact we have:

$$\begin{aligned} d_{n,k} &= [t^n]\frac{1}{1-t}\left(\frac{t}{1-t}\right)^k = [t^{n-k}]\frac{1}{(1-t)^{k+1}} = \\ &= \binom{-k-1}{n-k}(-1)^{n-k} = \binom{n}{n-k} = \binom{n}{k} \end{aligned}$$

and this shows that the generic element is actually a binomial coefficient. By formula (5.1.2), we find the well-known bivariate generating function $d(t,z) = (1-t-tz)^{-1}$.

From our point of view, one of the most important properties of Riordan arrays is the fact that the sums involving the rows of a Riordan array can be performed by operating a suitable transformation on a generating function and then by extracting a coefficient from the resulting function. More precisely, we prove the following theorem:

Theorem 5.1.1 *Let $D = (d(t), h(t))$ be a Riordan array and let $f(t)$ be the generating function of the sequence $(f_k)_{k\in N}$; then we have:*

$$\sum_{k=0}^{n} d_{n,k}f_k = [t^n]d(t)f(th(t)) \tag{5.1.3}$$

Proof: The proof consists in a straight-forward computation:

$$\begin{aligned} \sum_{k=0}^{n} d_{n,k}f_k &= \sum_{k=0}^{\infty} d_{n,k}f_k = \\ &= \sum_{k=0}^{\infty}[t^n]d(t)(th(t))^k f_k = \\ &= [t^n]d(t)\sum_{k=0}^{\infty} f_k(th(t))^k = \\ &= [t^n]d(t)f(th(t)). \end{aligned}$$

∎

In the case of Pascal triangle we obtain the Euler transformation:

$$\sum_{k=0}^{n}\binom{n}{k}f_k = [t^n]\frac{1}{1-t}f\left(\frac{t}{1-t}\right)$$

which we proved by simple considerations on binomial coefficients and generating functions. By considering the generating functions $\mathcal{G}(1) = 1/(1-t)$, $\mathcal{G}\big((-1)^k\big) = 1/(1+t)$ and $\mathcal{G}(k) = t/(1-t)^2$, from the previous theorem we immediately find:

$$\begin{aligned} &\text{row sums (r. s.)} && \sum_k d_{n,k} = [t^n]\frac{d(t)}{1-th(t)} \\ &\text{alternating r. s.} && \sum_k (-1)^k d_{n,k} = [t^n]\frac{d(t)}{1+th(t)} \\ &\text{weighted r. s.} && \sum_k k d_{n,k} = [t^n]\frac{td(t)h(t)}{(1-th(t))^2}. \end{aligned}$$

Moreover, by observing that $\widehat{D} = (d(t), th(t))$ is a Riordan array, whose rows are the diagonals of D,

we have:

$$\text{diagonal sums} \qquad \sum_k d_{n-k,k} = [t^n]\frac{d(t)}{1-t^2h(t)}.$$

Obviously, this observation can be generalized to find the generating function of any sum $\sum_k d_{n-sk,k}$ for every $s \geq 1$. We obtain well-known results for the Pascal triangle; for example, diagonal sums give:

$$\begin{aligned}\sum_k \binom{n-k}{k} &= [t^n]\frac{1}{1-t}\frac{1}{1-t^2(1-t)^{-1}} = \\ &= [t^n]\frac{1}{1-t-t^2} = F_{n+1}\end{aligned}$$

connecting binomial coefficients and Fibonacci numbers.

Another general result can be obtained by means of two sequences $(f_k)_{k\in N}$ and $(g_k)_{k\in N}$ and their generating functions $f(t), g(t)$. For $p = 1, 2, \ldots$, the generic element of the Riordan array $(f(t), t^{p-1})$ is:

$$d_{n,k} = [t^n]f(t)(t^p)^k = [t^{n-pk}]f(t) = f_{n-pk}.$$

Therefore, by formula (5.1.3), we have:

$$\sum_{k=0}^{\lfloor n/p \rfloor} f_{n-pk}g_k = [t^n]f(t)\big[g(y) \;\big|\; y = t^p\big] =$$

$$= [t^n]f(t)g(t^p).$$

This can be called the *rule of generalized convolution* since it reduces to the usual convolution rule for $p = 1$. Suppose, for example, that we wish to sum one out of every three powers of 2, starting with 2^n and going down to the lowest integer exponent ≥ 0; we have:

$$S_n = \sum_{k=0}^{\lfloor n/3 \rfloor} 2^{n-3k} = [t^n]\frac{1}{1-2t}\frac{1}{1-t^3}.$$

As we will learn studying asymptotics, an approximate value for this sum can be obtained by extracting the coefficient of the first factor and then by multiplying it by the second factor, in which t is substituted by $1/2$. This gives $S_n \approx 2^{n+3}/7$, and in fact we have the exact value $S_n = \lfloor 2^{n+3}/7 \rfloor$.

In a sense, the theorem on the sums involving the Riordan arrays is a characterization for them; in fact, we can prove a sort of inverse property:

Theorem 5.1.2 *Let $(d_{n,k})_{n,k\in N}$ be an infinite triangle such that for every sequence $(f_k)_{k\in N}$ we have $\sum_k d_{n,k}f_k = [t^n]d(t)f(th(t))$, where $f(t)$ is the generating function of the sequence and $d(t), h(t)$ are two f.p.s. not depending on $f(t)$. Then the triangle defined by the Riordan array $(d(t), h(t))$ coincides with $(d_{n,k})_{n,k\in N}$.*

Proof: For every $k \in \mathbb{N}$ take the sequence which is 0 everywhere except in the kth element $f_k = 1$. The corresponding generating function is $f(t) = t^k$ and we have $\sum_{i=0}^{\infty} d_{n,i}f_i = d_{n,k}$. Hence, according to the theorem's hypotheses, we find $\mathcal{G}_t\left(d_{n,k}\right)_{n,k\in N} = d_k(t) = d(t)(th(t))^k$, and this corresponds to the initial definition of column generating functions for a Riordan array, for every $k = 1, 2, \ldots$. ∎

5.2 The algebraic structure of Riordan arrays

The most important algebraic property of Riordan arrays is the fact that the usual row-by-column product of two Riordan arrays is a Riordan array. This is proved by considering two Riordan arrays $(d(t), h(t))$ and $(a(t), b(t))$ and performing the product, whose generic element is $\sum_j d_{n,j}f_{j,k}$, if $d_{n,j}$ is the generic element in $(d(t), h(t))$ and $f_{j,k}$ is the generic element in $(a(t), b(t))$. In fact we have:

$$\begin{aligned}&\sum_{j=0}^{\infty} d_{n,j}f_{j,k} = \\ &= \sum_{j=0}^{\infty}[t^n]d(t)(th(t))^j[y^j]a(y)(yb(y))^k = \\ &= [t^n]d(t)\sum_{j=0}^{\infty}(th(t))^j[y^j]a(y)(yb(y))^k = \\ &= [t^n]d(t)a(th(t))(th(t)b(th(t)))^k.\end{aligned}$$

By definition, the last expression denotes the generic element of the Riordan array $(f(t), g(t))$ where $f(t) = d(t)a(th(t))$ and $g(t) = h(t)b(th(t))$. Therefore we have:

$$(d(t), h(t)) \cdot (a(t), b(t)) = (d(t)a(th(t)), h(t)b(th(t))). \tag{5.2.1}$$

This expression is particularly important and is the basis for many developments of the Riordan array theory.

The product is obviously associative, and we observe that the Riordan array $(1, 1)$ acts as the neutral element or identity. In fact, the array $(1, 1)$ is everywhere 0 except for the elements on the main diagonal, which are 1. Observe that this array is proper.

Let us now suppose that $(d(t), h(t))$ is a proper Riordan array. By formula (5.2.1), we immediately see that the product of two proper Riordan arrays is proper; therefore, we can look for a proper Riordan array $(a(t), b(t))$ such that $(d(t), h(t)) \cdot (a(t), b(t)) = (1, 1)$. If this is the case, we should have:

$$d(t)a(th(t)) = 1 \quad \text{and} \quad h(t)b(th(t)) = 1.$$

By setting $y = th(t)$ we have:

$$a(y) = \left[d(t)^{-1} \;\middle|\; t = yh(t)^{-1} \right]$$

$$b(y) = \left[h(t)^{-1} \;\middle|\; t = yh(t)^{-1} \right].$$

Here we are in the hypotheses of the Lagrange Inversion Formula, and therefore there is a unique function $t = t(y)$ such that $t(0) = 0$ and $t = yh(t)^{-1}$. Besides, being $d(t), h(t) \in \mathcal{F}_0$, the two f.p.s. $a(y)$ and $b(y)$ are uniquely defined. We have therefore proved:

Theorem 5.2.1 *The set $\mathcal{A}$ of proper Riordan arrays is a group with the operation of row-by-column product defined functionally by relation (5.2.1).*

It is a simple matter to show that some important classes of Riordan arrays are subgroups of $\mathcal{A}$:

- the set of the Riordan arrays $(f(t), 1)$ is an invariant subgroup of $\mathcal{A}$; it is called the *Appell subgroup*;
- the set of the Riordan arrays $(1, g(t))$ is a subgroup of $\mathcal{A}$ and is called the *subgroup of associated operators* or the *Lagrange subgroup*;
- the set of Riordan arrays $(f(t), f(t))$ is a subgroup of $\mathcal{A}$ and is called the *Bell subgroup*. Its elements are also known as *renewal arrays*.

The first two subgroups have already been considered in the Chapter on "Formal Power Series" and show the connection between f.p.s. and Riordan arrays. The notations used in that Chapter are thus explained as particular cases of the most general case of (proper) Riordan arrays.

Let us now return to the formulas for a Riordan array inverse. If $h(t)$ is any fixed invertible f.p.s., let us define:

$$\overline{d}_h(t) = \left[d(y)^{-1} \;\middle|\; y = th(y)^{-1} \right]$$

so that we can write $(d(t), h(t))^{-1} = (\overline{d}_h(t), \overline{h}_h(t))$. By the product formula (5.2.1) we immediately find the identities:

$$d(t\overline{h}_h(t)) = \overline{d}_h(t)^{-1} \qquad \overline{d}_h(th(t)) = d(t)^{-1}$$

$$h(t\overline{h}_h(t)) = \overline{h}_h(t)^{-1} \qquad \overline{h}_h(th(t)) = h(t)^{-1}$$

which can be reduced to the single and basic rule:

$$f(t\overline{h}_h(t)) = \overline{f}_h(t)^{-1} \qquad \forall f(t) \in \mathcal{F}_0.$$

Observe that obviously $\overline{\overline{f}}_h(t) = f(t)$.

We wish now to find an explicit expression for the generic element $\overline{d}_{n,k}$ in the inverse Riordan array $(d(t), h(t))^{-1}$ in terms of $d(t)$ and $h(t)$. This will be done by using the LIF. As we observed in the first section, the bivariate generating function for $(d(t), h(t))$ is $d(t)/(1 - tzh(t))$ and therefore we have:

$$\begin{aligned} \overline{d}_{n,k} &= [t^n z^k] \frac{\overline{d}_h(t)}{1 - t\overline{z}\overline{h}_h(t)} = \\ &= [z^k][t^n] \left[\frac{\overline{d}_h(t)}{1 - zy} \;\middle|\; y = t\overline{h}_h(t) \right]. \end{aligned}$$

By the formulas above, we have:

$$y = t\overline{h}_h(t) = th(t\overline{h}_h(t))^{-1} = th(y)^{-1}$$

which is the same as $t = yh(y)$. Therefore we find: $\overline{d}_h(t) = \overline{d}_h(yh(y)) = d(t)^{-1}$, and consequently:

$$\begin{aligned} \overline{d}_{n,k} &= [z^k][t^n] \left[\frac{d(y)^{-1}}{1 - zy} \;\middle|\; y = th(y)^{-1} \right] = \\ &= [z^k] \frac{1}{n} [y^{n-1}] \left(\frac{d}{dy} \frac{d(y)^{-1}}{1 - zy} \right) \frac{1}{h(y)^n} = \\ &= [z^k] \frac{1}{n} [y^{n-1}] \left(\frac{z}{d(y)(1 - zy)^2} - \right. \\ &\quad \left. - \frac{d'(y)}{d(y)^2(1 - zy)} \right) \frac{1}{h(y)^n} = \\ &= [z^k] \frac{1}{n} [y^{n-1}] \left(\sum_{r=0}^{\infty} z^{r+1} y^r (r+1) - \right. \\ &\quad \left. - \frac{d'(y)}{d(y)} \sum_{r=0}^{\infty} z^r y^r \right) \frac{1}{d(y)h(y)^n} = \\ &= \frac{1}{n} [y^{n-1}] \left(k y^{k-1} - y^k \frac{d'(y)}{d(y)} \right) \frac{1}{d(y)h(y)^n} = \\ &= \frac{1}{n} [y^{n-k}] \left(k - \frac{y d'(y)}{d(y)} \right) \frac{1}{d(y)h(y)^n}. \end{aligned}$$

This is the formula we were looking for.

5.3 The A-sequence for proper Riordan arrays

Proper Riordan arrays play a very important role in our approach. Let us consider a Riordan array $D = (d(t), h(t))$, which is not proper, but $d(t) \in \mathcal{F}_0$. Since $h(0) = 0$, an $s > 0$ exists such that $h(t) = h_s t^s + h_{s+1} t^{s+1} + \cdots$ and $h_s \neq 0$. If we define $\widehat{h}(t) = h_s + h_{s+1} t + \cdots$, then $\widehat{h}(t) \in \mathcal{F}_0$. Consequently, the Riordan array $\widehat{D} = (d(t), \widehat{h}(t))$ is proper and the rows of D can be seen as the *s-diagonals* $(\widehat{d}_{n-sk})_{k \in N}$ of $\widehat{D}$. Fortunately, for proper Riordan arrays, Rogers has found an important characterization: every element $d_{n+1,k+1}$, $n, k \in \mathbb{N}$, can be expressed as a linear combination of the elements in the preceding row, i.e.:

$$d_{n+1,k+1} = a_0 d_{n,k} + a_1 d_{n,k+1} + a_2 d_{n,k+2} + \cdots =$$

$$= \sum_{j=0}^{\infty} a_j d_{n,k+j}. \tag{5.3.1}$$

The sum is actually finite and the sequence $A = (a_k)_{k\in N}$ is fixed. More precisely, we can prove the following theorem:

Theorem 5.3.1 *An infinite lower triangular array $D = (d_{n,k})_{n,k\in N}$ is a Riordan array if and only if a sequence $A = \{a_0 \neq 0, a_1, a_2, \ldots\}$ exists such that for every $n, k \in \mathbb{N}$ relation (5.3.1) holds*

Proof: Let us suppose that D is the Riordan array $(d(t), h(t))$ and let us consider the Riordan array $(d(t)h(t), h(t))$; we define the Riordan array $(A(t), B(t))$ by the relation:

$$(A(t), B(t)) = (d(t), h(t))^{-1} \cdot (d(t)h(t), h(t))$$

or:

$$(d(t), h(t)) \cdot (A(t), B(t)) = (d(t)h(t), h(t)).$$

By performing the product we find:

$$d(t)A(th(t)) = d(t)h(t) \quad \text{and} \quad h(t)B(th(t)) = h(t).$$

The latter identity gives $B(th(t)) = 1$ and this implies $B(t) = 1$. Therefore we have $(d(t), h(t)) \cdot (A(t), 1) = (d(t)h(t), h(t))$. The element $f_{n,k}$ of the left hand member is $\sum_{j=0}^{\infty} d_{n,j}a_{k-j} = \sum_{j=0}^{\infty} d_{n,k+j}a_j$, if as usual we interpret a_{k-j} as 0 when $k < j$. The same element in the right hand member is:

$$[t^n]d(t)h(t)(th(t))^k =$$

$$= [t^{n+1}]d(t)(th(t))^{k+1} = d_{n+1,k+1}.$$

By equating these two quantities, we have the identity (5.3.1). For the converse, let us observe that (5.3.1) uniquely defines the array D when the elements $\{d_{0,0}, d_{1,0}, d_{2,0}, \ldots\}$ of column 0 are given. Let $d(t)$ be the generating function of this column, $A(t)$ the generating function of the sequence A and define $h(t)$ as the solution of the functional equation $h(t) = A(th(t))$, which is uniquely determined because of our hypothesis $a_0 \neq 0$. We can therefore consider the proper Riordan array $\widehat{D} = (d(t), h(t))$; by the first part of the theorem, $\widehat{D}$ satisfies relation (5.3.1), for every $n, k \in \mathbb{N}$ and therefore, by our previous observation, it must coincide with D. This completes the proof. ∎

The sequence $A = (a_k)_{k\in N}$ is called the *A-sequence* of the Riordan array $D = (d(t), h(t))$ and it only depends on $h(t)$. In fact, as we have shown during the proof of the theorem, we have:

$$h(t) = A(th(t)) \tag{5.3.2}$$

and this uniquely determines A when $h(t)$ is given and, vice versa, $h(t)$ is uniquely determined when A is given.

The A-sequence for the Pascal triangle is the solution $A(y)$ of the functional equation $1/(1-t) = A(t/(1-t))$. The simple substitution $y = t/(1-t)$ gives $A(y) = 1 + y$, corresponding to the well-known basic recurrence of the Pascal triangle: $\binom{n+1}{k+1} = \binom{n}{k} + \binom{n}{k+1}$. At this point, we realize that we could have started with this recurrence relation and directly found $A(y) = 1+y$. Now, $h(t)$ is defined by (5.3.2) as the solution of $h(t) = 1+th(t)$, and this immediately gives $h(t) = 1/(1-t)$. Furthermore, since column 0 is $\{1, 1, 1, \ldots\}$, we have proved that the Pascal triangle corresponds to the Riordan array $(1/(1-t), 1/(1-t))$ as initially stated.

The pair of functions $d(t)$ and $A(t)$ completely characterize a proper Riordan array. Another type of characterization is obtained through the following observation:

Theorem 5.3.2 *Let $(d_{n,k})_{n,k\in N}$ be any infinite, lower triangular array with $d_{n,n} \neq 0, \forall n \in \mathbb{N}$ (in particular, let it be a proper Riordan array); then a unique sequence $Z = (z_k)_{k\in N}$ exists such that every element in column 0 can be expressed as a linear combination of all the elements in the preceding row, i.e.:*

$$d_{n+1,0} = z_0 d_{n,0} + z_1 d_{n,1} + z_2 d_{n,2} + \cdots = \sum_{j=0}^{\infty} z_j d_{n,j}. \tag{5.3.3}$$

Proof: Let $z_0 = d_{1,0}/d_{0,0}$. Now we can uniquely determine the value of z_1 by expressing $d_{2,0}$ in terms of the elements in row 1, i.e.:

$$d_{2,0} = z_0 d_{1,0} + z_1 d_{1,1} \quad \text{or} \quad z_1 = \frac{d_{0,0}d_{2,0} - d_{1,0}^2}{d_{0,0}d_{1,1}}.$$

In the same way, we determine z_2 by expressing $d_{3,0}$ in terms of the elements in row 2, and by substituting the values just obtained for z_0 and z_1. By proceeding in the same way, we determine the sequence Z in a unique way. ∎

The sequence Z is called the *Z-sequence* for the (Riordan) array; it characterizes column 0, except for the element $d_{0,0}$. Therefore, we can say that the triple $(d_{0,0}, A(t), Z(t))$ completely characterizes a proper Riordan array. To see how the Z-sequence is obtained by starting with the usual definition of a Riordan array, let us prove the following:

Theorem 5.3.3 *Let $(d(t), h(t))$ be a proper Riordan array and let $Z(t)$ be the generating function of the array's Z-sequence. We have:*

$$d(t) = \frac{d_{0,0}}{1 - tZ(th(t))}$$

Proof: By the preceding theorem, the Z-sequence exists and is unique. Therefore, equation (5.3.3) is valid for every $n \in \mathbb{N}$, and we can go on to the generating functions. Since $d(t)(th(t))^k$ is the generating function for column k, we have:

$$\begin{aligned}
&\frac{d(t) - d_{0,0}}{t} = \\
&= z_0 d(t) + z_1 d(t) th(t) + z_2 d(t)(th(t))^2 + \cdots = \\
&= d(t)(z_0 + z_1 th(t) + z_2 (th(t))^2 + \cdots) = \\
&= d(t) Z(th(t)).
\end{aligned}$$

By solving this equation in $d(t)$, we immediately find the relation desired. ∎

The relation can be inverted and this gives us the formula for the Z-sequence:

$$Z(y) = \left[\frac{d(t) - d_{0,0}}{t d(t)} \;\middle|\; t = yh(t)^{-1} \right].$$

We conclude this section by giving a theorem, which characterizes renewal arrays by means of the A- and Z-sequences:

Theorem 5.3.4 *Let $d(0) = h(0) \neq 0$. Then $d(t) = h(t)$ if and only if: $A(y) = d(0) + yZ(y)$.*

Proof: Let us assume that $A(y) = d(0) + yZ(y)$ or $Z(y) = (A(y) - d(0))/y$. By the previous theorem, we have:

$$\begin{aligned}
d(t) &= \frac{d(0)}{1 - tZ(th(t))} = \\
&= \frac{d(0)}{1 - (tA(th(t)) - d(0)t)/th(t)} = \\
&= \frac{d(0) th(t)}{d(0) t} = h(t),
\end{aligned}$$

because $A(th(t)) = h(t)$ by formula (5.3.2). Vice versa, by the formula for $Z(y)$, we obtain from the hypothesis $d(t) = h(t)$:

$$\begin{aligned}
&d(0) + yZ(y) = \\
&= \left[d(0) + y\left(\frac{1}{t} - \frac{d(0)}{th(t)}\right) \;\middle|\; t = yh(t)^{-1} \right] = \\
&= \left[d(0) + \frac{th(t)}{t} - \frac{d(0) th(t)}{th(t)} \;\middle|\; t = yh(t)^{-1} \right] = \\
&= \left[h(t) \;\middle|\; t = yh(t)^{-1} \right] = A(y).
\end{aligned}$$

∎

5.4 Simple binomial coefficients

Let us consider *simple binomial coefficients*, i.e., binomial coefficients of the form $\binom{n+ak}{m+bk}$, where a, b are two parameters and k is a non-negative integer variable. Depending if we consider n a variable and m a parameter, or vice versa, we have two different infinite arrays $(d_{n,k})$ or $(\widehat{d}_{m,k})$, whose elements depend on the parameters a, b, m or a, b, n, respectively. In either case, if some conditions on a, b hold, we have Riordan arrays and therefore we can apply formula (5.1.3) to find the value of many sums.

Theorem 5.4.1 *Let $d_{n,k}$ and $\widehat{d}_{m,k}$ be as above. If $b > a$ and $b - a$ is an integer, then $D = (d_{n,k})$ is a Riordan array. If $b < 0$ is an integer, then $\widehat{D} = (\widehat{d}_{m,k})$ is a Riordan array. We have:*

$$D = \left(\frac{t^m}{(1-t)^{m+1}}, \frac{t^{b-a-1}}{(1-t)^b} \right)$$

$$\widehat{D} = \left((1+t)^n, \frac{t^{-b-1}}{(1+t)^{-a}} \right).$$

Proof: By using well-known properties of binomial coefficients, we find:

$$\begin{aligned}
d_{n,k} &= \binom{n+ak}{m+bk} = \binom{n+ak}{n-m+ak-bk} = \\
&= \binom{-n-ak+n-m+ak-bk-1}{n-m+ak-bk} \times \\
&\quad \times (-1)^{n-m+ak-bk} = \\
&= \binom{-m-bk-1}{(n-m)+(a-b)k} (-1)^{n-m+ak-bk} = \\
&= [t^{n-m+ak-bk}] \frac{1}{(1-t)^{m+1+bk}} = \\
&= [t^n] \frac{t^m}{(1-t)^{m+1}} \left(\frac{t^{b-a}}{(1-t)^b} \right)^k ;
\end{aligned}$$

and:

$$\begin{aligned}
\widehat{d}_{m,k} &= [t^{m+bk}](1+t)^{n+ak} = \\
&= [t^m](1+t)^n (t^{-b}(1+t)^a)^k.
\end{aligned}$$

The theorem now directly follows from (5.1.1) ∎

For $m = a = 0$ and $b = 1$ we again find the Riordan array of the Pascal triangle. The sum (5.1.3) takes on two specific forms which are worth being stated explicitly:

$$\sum_k \binom{n+ak}{m+bk} f_k =$$

$$= [t^n] \frac{t^m}{(1-t)^{m+1}} f\left(\frac{t^{b-a}}{(1-t)^b} \right) \qquad b > a \qquad (5.4.1)$$

$$\sum_k \binom{n+ak}{m+bk} f_k =$$

$$= [t^m](1+t)^n f(t^{-b}(1+t)^a) \qquad b < 0. \qquad (5.4.2)$$

If m and n are independent of each other, these relations can also be stated as generating function identities. The binomial coefficient $\binom{n+ak}{m+bk}$ is so general that a large number of combinatorial sums can be solved by means of the two formulas (5.4.1) and (5.4.2).

Let us begin our set of examples with a simple sum; by the theorem above, the binomial coefficients $\binom{n-k}{m}$ corresponds to the Riordan array $(t^m/(1+t)^{m+1}, 1)$; therefore, by the formula concerning the row sums, we have:

$$\begin{aligned}\sum_k \binom{n-k}{m} &= [t^n]\frac{t^m}{(1-t)^{m+1}}\frac{1}{1-t} = \\ &= [t^{n-m}]\frac{1}{(1-t)^{m+2}} = \binom{n+1}{m+1}.\end{aligned}$$

Another simple example is the sum:

$$\begin{aligned}&\sum_k \binom{n}{2k+1}5^k = \\ &= [t^n]\frac{t}{(1-t)^2}\left[\frac{1}{1-5y}\;\Big|\; y=\frac{t^2}{(1-t)^2}\right] = \\ &= \frac{1}{2}[t^n]\frac{2t}{1-2t-4t^2} = 2^{n-1}F_n.\end{aligned}$$

The following sum is a more interesting case. From the generating function of the Catalan numbers we immediately find:

$$\begin{aligned}&\sum_k \binom{n+k}{m+2k}\binom{2k}{k}\frac{(-1)^k}{k+1} = \\ &= [t^n]\frac{t^m}{(1-t)^{m+1}}\left[\frac{\sqrt{1+4y}-1}{2y}\;\Big|\; y=\frac{t}{(1-t)^2}\right] = \\ &= [t^{n-m}]\frac{1}{(1-t)^{m+1}}\left(\sqrt{1+\frac{4t}{(1-t)^2}}-1\right)\times \\ &\quad\times\frac{(1-t)^2}{2t} = \\ &= [t^{n-m}]\frac{1}{(1-t)^m} = \binom{n-1}{m-1}.\end{aligned}$$

In the following sum we use the bisection formulas. Because the generating function for $\binom{z+1}{k}2^k$ is $(1+2t)^{z+1}$, we have:

$$\begin{aligned}&\mathcal{G}\left(\binom{z+1}{2k+1}2^{2k+1}\right) = \\ &= \frac{1}{2\sqrt{t}}\left((1+2\sqrt{t})^{z+1}-(1-2\sqrt{t})^{z+1}\right).\end{aligned}$$

By applying formula (5.4.2):

$$\begin{aligned}&\sum_k \binom{z+1}{2k+1}\binom{z-2k}{n-k}2^{2k+1} = \\ &= [t^n](1+t)^z\left[\frac{(1+2\sqrt{y})^{z+1}}{2\sqrt{y}}-\right. \\ &\quad\left.-\frac{(1-2\sqrt{y})^{z+1}}{2\sqrt{y}}\;\Big|\; y=\frac{t}{(1+t)^2}\right] = \\ &= [t^n](1+t)^{z+1}\left(\frac{(1+t+2\sqrt{t})^{z+1}}{2\sqrt{t}(1+t)^{z+1}}-\right. \\ &\quad\left.-\frac{(1+t-2\sqrt{t})^{z+1}}{2\sqrt{t}(1+t)^{z+1}}\right) = \\ &= [t^{2n+1}](1+t)^{2z+2} = \binom{2z+2}{2n+1};\end{aligned}$$

in the last but one passage, we used backwards the bisection rule, since $(1+t\pm 2\sqrt{t})^{z+1} = (1\pm\sqrt{t})^{2z+2}$.

We solve the following sum by using (5.4.2):

$$\begin{aligned}&\sum_k \binom{2n-2k}{m-k}\binom{n}{k}(-2)^k = \\ &= [t^m](1+t)^{2n}\left[(1-2y)^n\;\Big|\; y=\frac{t}{(1-t)^2}\right] = \\ &= [t^m](1+t^2)^n = \binom{n}{m/2}\end{aligned}$$

where the binomial coefficient is to be taken as zero when m is odd.

5.5 Other Riordan arrays from binomial coefficients

Other Riordan arrays can be found by using the theorem in the previous section and the rule ($\alpha \neq 0$, if $-$ is considered):

$$\frac{\alpha\pm\beta}{\alpha}\binom{\alpha}{\beta} = \binom{\alpha}{\beta}\pm\binom{\alpha-1}{\beta-1}.$$

For example we find:

$$\begin{aligned}\frac{2n}{n+k}\binom{n+k}{2k} &= \frac{2n}{n+k}\binom{n+k}{n-k} = \\ &= \binom{n+k}{n-k}+\binom{n+k-1}{n-k-1} = \\ &= \binom{n+k}{2k}+\binom{n-1+k}{2k}.\end{aligned}$$

Hence, by formula (5.4.1), we have:

$$\begin{aligned}&\sum_k \frac{2n}{n+k}\binom{n+k}{n-k}f_k = \\ &= \sum_k \binom{n+k}{2k}f_k+\sum_k\binom{n-1+k}{2k}f_k = \\ &= [t^n]\frac{1}{1-t}f\left(\frac{t}{(1-t)^2}\right)+ \\ &\quad+[t^{n-1}]\frac{1}{1-t}f\left(\frac{t}{(1-t)^2}\right) = \\ &= [t^n]\frac{1+t}{1-t}f\left(\frac{t}{(1-t)^2}\right).\end{aligned}$$

This proves that the infinite triangle of the elements $\frac{2n}{n+k}\binom{n+k}{2k}$ is a proper Riordan array and many identities can be proved by means of the previous formula. For example:

$$\begin{aligned}\sum_k \frac{2n}{n+k}\binom{n+k}{n-k}\binom{2k}{k}(-1)^k = \\ = \quad [t^n]\frac{1+t}{1-t}\left[\frac{1}{\sqrt{1+4y}}\;\Big|\; y=\frac{t}{(1-t)^2}\right] = \\ = \quad [t^n]1 = \delta_{n,0},\end{aligned}$$

$$\begin{aligned}\sum_k \frac{2n}{n+k}\binom{n+k}{n-k}\binom{2k}{k}\frac{(-1)^k}{k+1} = \\ = \quad [t^n]\frac{1+t}{1-t}\left[\frac{\sqrt{1+4y}-1}{2y}\;\Big|\; y=\frac{t}{(1-t)^2}\right] = \\ = \quad [t^n](1+t) = \delta_{n,0}+\delta_{n,1}.\end{aligned}$$

The following is a quite different case. Let $f(t) = \mathcal{G}(f_k)$ and:

$$G(t) = \mathcal{G}\left(\frac{f_k}{k}\right) = \int_0^t \frac{f(\tau)-f_0}{\tau}d\tau.$$

Obviously we have:

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

except for $k=0$, when the left-hand side is 1 and the right-hand side is not defined. By formula (5.4.1):

$$\begin{aligned}\sum_k \frac{n}{n-k}\binom{n-k}{k}f_k = \\ = \quad f_0 + n\sum_{k=1}^{\infty}\binom{n-k-1}{k-1}\frac{f_k}{k} = \\ = \quad f_0 + n[t^n]G\left(\frac{t^2}{1-t}\right). \qquad (5.5.1)\end{aligned}$$

This gives an immediate proof of the following formula known as *Hardy's identity*:

$$\begin{aligned}\sum_k \frac{n}{n-k}\binom{n-k}{k}(-1)^k = \\ = \quad [t^n]\ln\frac{1-t^2}{1+t^3} = \\ = \quad [t^n]\left(\ln\frac{1}{1+t^3} - \ln\frac{1}{1-t^2}\right) = \\ = \quad \begin{cases}(-1)^n 2/n & \text{if 3 divides } n\\ (-1)^{n-1}/n & \text{otherwise.}\end{cases}\end{aligned}$$

We also immediately obtain:

$$\sum_k \frac{1}{n-k}\binom{n-k}{k} = \frac{\phi^n + \widehat{\phi}^n}{n}$$

where ϕ is the golden ratio and $\widehat{\phi} = -\phi^{-1}$. The reader can generalize formula (5.5.1) by using the change of variable $t \to pt$ and prove other formulas. The following one is known as *Riordan's old identity*:

$$\sum_k \frac{n}{n-k}\binom{n-k}{k}(a+b)^{n-2k}(-ab)^k = a^n + b^n$$

while this is a generalization of Hardy's identity:

$$\begin{aligned}\sum_k \frac{n}{n-k}\binom{n-k}{k}x^{n-2k}(-1)^k = \\ = \frac{(x+\sqrt{x^2-4})^n + (x-\sqrt{x^2-4})^n}{2^n}.\end{aligned}$$

5.6 Binomial coefficients and the LIF

In a few cases only, the formulas of the previous sections give the desired result when the m and n in the numerator and denominator of a binomial coefficient are related between them. In fact, in that case, we have to extract the coefficient of t^n from a function depending on the same variable n (or m). This requires to apply the Lagrange Inversion Formula, according to the diagonalization rule. Let us suppose we have the binomial coefficient $\binom{2n-k}{n-k}$ and we wish to know whether it corresponds to a Riordan array or not. We have:

$$\begin{aligned}\binom{2n-k}{n-k} &= [t^{n-k}](1+t)^{2n-k} = \\ &= [t^n](1+t)^{2n}\left(\frac{t}{1+t}\right)^k.\end{aligned}$$

The function $(1+t)^{2n}$ cannot be assumed as the $d(t)$ function of a Riordan array because it varies as n varies. Therefore, let us suppose that k is fixed; we can apply the diagonalization rule with $F(t) = (t/(1+t))^k$ and $\phi(t) = (1+t)^2$, and try to find a true generating function. We have to solve the equation:

$$w = t\phi(w) \qquad \text{or} \qquad w = t(1+w)^2.$$

This equation is $tw^2 - (1-2t)w + t = 0$ and we are looking for the unique solution $w = w(t)$ such that $w(0) = 0$. This is:

$$w(t) = \frac{1-2t-\sqrt{1-4t}}{2t}.$$

We now perform the necessary computations:

$$F(w) \quad = \quad \left(\frac{w}{1+w}\right)^k =$$

$$= \left(\frac{1-2t-\sqrt{1-4t}}{1-\sqrt{1-4t}}\right)^k = \\ = \left(\frac{1-\sqrt{1-4t}}{2}\right)^k;$$

furthermore:

$$\frac{1}{1-t\phi'(w)} = \frac{1}{1-2t(1+w)} = \frac{1}{\sqrt{1-4t}}.$$

Therefore, the diagonalization gives:

$$\binom{2n-k}{n-k} = [t^n]\frac{1}{\sqrt{1-4t}}\left(\frac{1-\sqrt{1-4t}}{2}\right)^k.$$

This shows that the binomial coefficient is the generic element of the Riordan array:

$$D = \left(\frac{1}{\sqrt{1-4t}}, \frac{1-\sqrt{1-4t}}{2t}\right).$$

As a check, we observe that column 0 contains all the elements with $k=0$, i.e., $\binom{2n}{n}$, and this is in accordance with the generating function $d(t) = 1/\sqrt{1-4t}$. A simple example is:

$$\sum_{k=0}^{n}\binom{2n-k}{n-k}2^k = \\ = [t^n]\frac{1}{\sqrt{1-4t}}\left[\frac{1}{1-2y}\;\middle|\; y=\frac{1-\sqrt{1-4t}}{2}\right] = \\ = [t^n]\frac{1}{\sqrt{1-4t}}\frac{1}{\sqrt{1-4t}} = [t^n]\frac{1}{1-4t} = 4^n.$$

By using the diagonalization rule as above, we can show that:

$$\left(\binom{2n+ak}{n-ck}\right)_{k\in N} = \\ = \left(\frac{1}{\sqrt{1-4t}}, t^{c-1}\left(\frac{1-\sqrt{1-4t}}{2t}\right)^{a+2c}\right).$$

An interesting example is given by the following alternating sum:

$$\sum_k \binom{2n}{n-3k}(-1)^k = \\ = [t^n]\frac{1}{\sqrt{1-4t}}\left[\frac{1}{1+y}\;\middle|\; y=t^3\left(\frac{1-\sqrt{1-4t}}{2t}\right)^6\right] \\ = [t^n]\left(\frac{1}{2\sqrt{1-4t}}+\frac{1-t}{2(1-3t)}\right) = \\ = \frac{1}{2}\binom{2n}{n} + 3^{n-1} + \frac{\delta_{n,0}}{6}.$$

The reader is invited to solve, in a similar way, the corresponding non-alternating sum.

In the same way we can deal with binomial coefficients of the form $\binom{pn+ak}{n-ck}$, but in this case, in order to apply the LIF, we have to solve an equation of degree $p > 2$. This creates many difficulties, and we do not insist on it any longer.

5.7 Coloured walks

In the section "Walks, trees and Catalan numbers" we introduced the concept of a walk or path on the integral lattice $\mathbb{Z}^2$. The concept can be generalized by defining a walk as a sequence of steps starting from the origin and composed by three kinds of steps:

1. *east steps*, which go from the point (x, y) to $(x+1, y)$;
2. *diagonal steps*, which go from the point (x, y) to $(x+1, y+1)$;
3. *north steps*, which go from the point (x, y) to $(x, y+1)$.

A *colored walk* is a walk in which every kind of step can assume different colors; we denote by a, b, c ($a > 0, b, c \geq 0$) the number of colors the east, diagonal and north steps can be. We discuss *complete* colored walks, i.e., walks without any restriction, and *under-diagonal* walks, i.e., walks that never go above the main diagonal $x - y = 0$. The *length* of a walk is the number of its steps, and we denote by $d_{n,k}$ the number of colored walks which have length n and reach a distance k from the main diagonal, i.e., the last step ends on the diagonal $x - y = k \geq 0$. A *colored walk problem* is any (counting) problem corresponding to colored walks; a problem is called *symmetric* if and only if $a = c$.

We wish to point out that our considerations are by no means limited to the walks on the integral lattice. Many combinatorial problems can be proved to be equivalent to some walk problems; bracketing problems are a typical example and, in fact, a vast literature exists on walk problems.

Let us consider $d_{n+1,k+1}$, i.e., the number of colored walks of length $n+1$ reaching the distance $k+1$ from the main diagonal. We observe that each walk is obtained in a unique way as:

1. a walk of length n reaching the distance k from the main diagonal, followed by any of the a east steps;
2. a walk of length n reaching the distance $k+1$ from the main diagonal, followed by any of the b diagonal steps;
3. a walk of length n reaching the distance $k+2$ from the main diagonal, followed by any of the c north steps.

Hence we have: $d_{n+1,k+1} = ad_{n,k} + bd_{n,k+1} + cd_{n,k+2}$. This proves that $A = \{a, b, c\}$ is the A-sequence of $(d_{n,k})_{n,k\in N}$, which therefore is a proper Riordan array. This significant fact can be stated as:

Theorem 5.7.1 *Let $d_{n,k}$ be the number of colored walks of length n reaching a distance k from the main diagonal, then the infinite triangle $(d_{n,k})_{n,k\in N}$ is a proper Riordan array.*

The Pascal, Catalan and Motzkin triangles define walking problems that have different values of a, b, c. When $c = 0$, it is easily proved that $d_{n,k} = \binom{n}{k} a^k b^{n-k}$ and so we end up with the Pascal triangle. Consequently, we assume $c \neq 0$. For any given triple (a, b, c) we obtain one type of array from complete walks and another from underdiagonal walks. However, the function $h(t)$, that only depends on the A-sequence, is the same in both cases, and we can find it by means of formula (5.3.2). In fact, $A(t) = a+bt+ct^2$ and $h(t)$ is the solution of the functional equation $h(t) = a + bth(t) + ct^2h(t)^2$ having $h(0) \neq 0$:

$$h(t) = \frac{1 - bt - \sqrt{1 - 2bt + b^2t^2 - 4act^2}}{2ct^2} \qquad (5.7.1)$$

The radicand $1 - 2bt + (b^2 - 4ac)t^2 = (1 - (b + 2\sqrt{ac})t)(1 - (b - 2\sqrt{ac})t)$ will be simply denoted by Δ.

Let us now focus our attention on underdiagonal walks. If we consider $d_{n+1,0}$, we observe that every walk returning to the main diagonal can only be obtained from another walk returning to the main diagonal followed by any diagonal step, or a walk ending at distance 1 from the main diagonal followed by any north step. Hence, we have $d_{n+1,0} = bd_{n,0} + cd_{n,1}$ and in the column generating functions this corresponds to $d(t) - 1 = btd(t) + ctd(t)th(t)$. From this relation we easily find $d(t) = (1/a)h(t)$, and therefore by (5.7.1) the Riordan array of underdiagonal colored walk is:

$$(d_{n,k})_{n,k\in N} = \left(\frac{1 - bt - \sqrt{\Delta}}{2act^2}, \frac{1 - bt - \sqrt{\Delta}}{2ct^2} \right).$$

In current literature, major importance is usually given to the following three quantities:

1. the number of walks returning to the main diagonal; this is $d_n = [t^n]d(t)$, for every n,
2. the total number of walks of length n; this is $\alpha_n = \sum_{k=0}^{n} d_{n,k}$, i.e., the value of the row sums of the Riordan array;
3. the average distance from the main diagonal of all the walks of length n; this is $\delta_n = \sum_{k=0}^{n} kd_{n,k}$, which is the weighted row sum of the Riordan array, divided by α_n.

In Chapter 7 we will learn how to find an asymptotic approximation for d_n. With regard to the last two points, the formulas for the row sums and the weighted row sums given in the first section allow us to find the generating functions $\alpha(t)$ of the total number α_n of underdiagonal walks of length n, and $\delta(t)$ of the total distance δ_n of these walks from the main diagonal:

$$\alpha(t) = \frac{1}{2at} \frac{1 - (b + 2a)t - \sqrt{\Delta}}{(a + b + c)t - 1}$$

$$\delta(t) = \frac{1}{4at} \left(\frac{1 - (b + 2a)t - \sqrt{\Delta}}{(a + b + c)t - 1} \right)^2.$$

In the symmetric case these formulas simplify as follows:

$$\alpha(t) = \frac{1}{2at} \left(\sqrt{\frac{1 - (b - 2a)t}{1 - (b + 2a)t}} - 1 \right)$$

$$\delta(t) = \frac{1}{2at} \left(\frac{1 - bt}{1 - (b + 2a)t} - \sqrt{\frac{1 - (b - 2a)t}{1 - (b + 2a)t}} \right).$$

The alternating row sums and the diagonal sums sometimes have some combinatorial significance as well, and so they can be treated in the same way.

The study of complete walks follows the same lines and we only have to derive the form of the corresponding Riordan array, which is:

$$(d_{n,k})_{n,k\in N} = \left(\frac{1}{\sqrt{\Delta}}, \frac{1 - bt - \sqrt{\Delta}}{2ct^2} \right).$$

The proof is as follows. Since a complete walk can go above the main diagonal, the array $(d_{n,k})_{n,k\in N}$ is only the right part of an infinite triangle, in which k can also assume the negative values. By following the logic of the theorem above, we see that the generating function of the nth row is $((c/w) + b + aw)^n$, and therefore the bivariate generating function of the extended triangle is:

$$\begin{aligned} d(t, w) &= \sum_n \left(\frac{c}{w} + b + aw \right)^n t^n = \\ &= \frac{1}{1 - (aw + b + c/w)t}. \end{aligned}$$

If we expand this expression by partial fractions, we get:

$$\begin{aligned} d(t, w) &= \frac{1}{\sqrt{\Delta}} \left(\frac{1}{1 - \frac{1-bt-\sqrt{\Delta}}{2ct} w} - \frac{1}{1 - \frac{1-bt+\sqrt{\Delta}}{2ct} w} \right) \\ &= \frac{1}{\sqrt{\Delta}} \left(\frac{1}{1 - \frac{1-bt-\sqrt{\Delta}}{2ct} w} + \right. \\ &\quad \left. + \frac{1 - bt - \sqrt{\Delta}}{2at} \frac{1}{w} \frac{1}{1 - \frac{1-bt-\sqrt{\Delta}}{2ct} \frac{1}{w}} \right). \end{aligned}$$

The first term represents the right part of the extended triangle and this corresponds to $k \geq 0$, whereas the second term corresponds to the left part ($k < 0$). We are interested in the right part, and the expression can be written as:

$$\frac{1}{\sqrt{\Delta}} \frac{1}{1 - \frac{1-bt-\sqrt{\Delta}}{2ct} w} = \frac{1}{\sqrt{\Delta}} \sum_k \left(\frac{1 - bt - \sqrt{\Delta}}{2ct} \right)^k w^k$$

which immediately gives the form of the Riordan array.

5.8 Stirling numbers and Riordan arrays

The connection between Riordan arrays and Stirling numbers is not immediate. If we examine the two infinite triangles of the Stirling numbers of both kinds, we immediately realize that they are *not* Riordan arrays. It is not difficult to obtain the column generating functions for the Stirling numbers of the second kind; by starting with the recurrence relation:

$$\left\{ {n+1 \atop k+1} \right\} = (k+1) \left\{ {n \atop k+1} \right\} + \left\{ {n \atop k} \right\}$$

and the obvious generating function $S_0(t) = 1$, we can specialize the recurrence (valid for every $n \in \mathbb{N}$) to the case $k = 0$. This gives the relation between generating functions:

$$\frac{S_1(t) - S_1(0)}{t} = S_1(t) + S_0(t);$$

because $S_1(0) = 0$, we immediately obtain $S_1(t) = t/(1-t)$, which is easily checked by looking at column 1 in the array. In a similar way, by specializing the recurrence relation to $k = 1$, we find $S_2(t) = 2tS_2(t) + tS_1(t)$, whose solution is:

$$S_2(t) = \frac{t^2}{(1-t)(1-2t)}.$$

This proves, in an algebraic way, that $\left\{ {n \atop 2} \right\} = 2^{n-1} - 1$, and also indicates the form of the generating function for column m:

$$S_m(t) = \mathcal{G}\left(\left\{ {n \atop m} \right\} \right)_{n \in N} = \frac{t^m}{(1-t)(1-2t)\cdots(1-mt)}$$

which is now proved by induction when we specialize the recurrence relation above to $k = m$. This is left to the reader as a simple exercise.

The generating functions for the Stirling numbers of the first kind are not so simple. However, let us go on with the Stirling numbers of the second kind proceeding in the following way; if we multiply the recurrence relation by $(k+1)!/(n+1)!$ we obtain the new relation:

$$\frac{(k+1)!}{(n+1)!} \left\{ {n+1 \atop k+1} \right\} =$$

$$= \frac{(k+1)!}{n!} \left\{ {n \atop k+1} \right\} \frac{k+1}{n+1} + \frac{k!}{n!} \left\{ {n \atop k} \right\} \frac{k+1}{n+1}.$$

If we denote by $d_{n,k}$ the quantity $k! \left\{ {n \atop k} \right\} / n!$, this is a recurrence relation for $d_{n,k}$, which can be written as:

$$(n+1) d_{n+1,k+1} = (k+1) d_{n,k+1} + (k+1) d_{n,k}.$$

Let us now proceed as above and find the column generating functions for the new array $(d_{n,k})_{n,k \in N}$. Obviously, $d_0(t) = 1$; by setting $k = 0$ in the new recurrence:

$$(n+1) d_{n+1,1} = d_{n,1} + d_{n,0}$$

and passing to generating functions: $d_1'(t) = d_1(t) + 1$. The solution of this simple differential equation is $d_1(t) = e^t - 1$ (the reader can simply check this solution, if he or she prefers). We can now go on by setting $k = 1$ in the recurrence; we obtain: $(n+1) d_{n+1,2} = 2d_{n,2} + 2d_{n,1}$, or $d_2'(t) = 2d_2(t) + 2(e^t - 1)$. Again, this differential equation has the solution $d_2(t) = (e^t - 1)^2$, and this suggests that, in general, we have: $d_k(t) = (e^t - 1)^k$. A rigorous proof of this fact can be obtained by mathematical induction; the recurrence relation gives: $d_{k+1}'(t) = (k+1) d_{k+1}(t) + (k+1) d_k(t)$. By the induction hypothesis, we can substitute $d_k(t) = (e^t - 1)^k$ and solve the differential equation thus obtained. In practice, we can simply verify that $d_{k+1}(t) = (e^t - 1)^{k+1}$; by substituting, we have:

$$(k+1) e^t (e^t - 1)^k =$$

$$(k+1)(e^t - 1)^{k+1} + (k+1)(e^t - 1)^k$$

and this equality is obviously true.

The form of this generating function:

$$d_k(t) = \mathcal{G}\left(\frac{k!}{n!} \left\{ {n \atop k} \right\} \right)_{n \in N} = (e^t - 1)^k$$

proves that $(d_{n,k})_{n,k \in N}$ is a Riordan array having $d(t) = 1$ and $th(t) = (e^t - 1)$. This fact allows us to prove algebraically a lot of identities concerning the Stirling numbers of the second kind, as we shall see in the next section.

For the Stirling numbers of the first kind we proceed in an analogous way. We multiply the basic recurrence:

$$\left[{n+1 \atop k+1} \right] = n \left[{n \atop k+1} \right] + \left[{n \atop k} \right]$$

by $(k+1)!/(n+1)!$ and study the quantity $f_{n,k} = k!\begin{bmatrix} n \\ k \end{bmatrix}/n!$:

$$\frac{(k+1)!}{(n+1)!}\begin{bmatrix} n+1 \\ k+1 \end{bmatrix} =$$

$$= \frac{(k+1)!}{n!}\begin{bmatrix} n \\ k+1 \end{bmatrix}\frac{n}{n+1} + \frac{k!}{n!}\begin{bmatrix} n \\ k \end{bmatrix}\frac{k+1}{n+1},$$

that is:

$$(n+1)f_{n+1,k+1} = nf_{n,k+1} + (k+1)f_{n,k}.$$

In this case also we have $f_0(t) = 1$ and by specializing the last relation to the case $k = 0$, we obtain:

$$f_1'(t) = tf_1'(t) + f_0(t).$$

This is equivalent to $f_1'(t) = 1/(1-t)$ and because $f_1(0) = 0$ we have:

$$f_1(t) = \ln\frac{1}{1-t}.$$

By setting $k = 1$, we find the simple differential equation $f_2'(t) = tf_2'(t) + 2f_1(t)$, whose solution is:

$$f_2(t) = \left(\ln\frac{1}{1-t}\right)^2.$$

This suggests the general formula:

$$f_k(t) = \mathcal{G}\left(\frac{k!}{n!}\begin{bmatrix} n \\ k \end{bmatrix}\right)_{n\in N} = \left(\ln\frac{1}{1-t}\right)^k$$

and again this can be proved by induction. In this case, $(f_{n,k})_{n,k\in N}$ is the Riordan array having $d(t) = 1$ and $th(t) = \ln(1/(1-t))$.

5.9 Identities involving the Stirling numbers

The two recurrence relations for $d_{n,k}$ and $f_{n,k}$ do not give an immediate evidence that the two triangles are indeed Riordan arrays., because they do not correspond to A-sequences. However, the A-sequences for the two arrays can be easily found, once we know their $h(t)$ function. For the Stirling numbers of the first kind we have to solve the functional equation:

$$\ln\frac{1}{1-t} = tA\left(\ln\frac{1}{1-t}\right).$$

By setting $y = \ln(1/(1-t))$ or $t = (e^y-1)/y$, we have $A(y) = ye^y/(e^y-1)$ and this is the generating function for the A-sequence we were looking for. In a similar way, we find that the A-sequence for the triangle related to the Stirling numbers of the second kind is:

$$A(t) = \frac{t}{\ln(1+t)}.$$

A first result we obtain by using the correspondence between Stirling numbers and Riordan arrays concerns the row sums of the two triangles. For the Stirling numbers of the first kind we have:

$$\begin{aligned}\sum_{k=0}^{n}\begin{bmatrix} n \\ k \end{bmatrix} &= n!\sum_{k=0}^{n}\frac{k!}{n!}\begin{bmatrix} n \\ k \end{bmatrix}\frac{1}{k!} = \\ &= n![t^n]\left[e^y \;\middle|\; y = \ln\frac{1}{1-t}\right] = \\ &= n![t^n]\frac{1}{1-t} = n!\end{aligned}$$

as we observed and proved in a combinatorial way. The row sums of the Stirling numbers of the second kind give, as we know, the Bell numbers; thus we can obtain the (exponential) generating function for these numbers:

$$\begin{aligned}\sum_{k=0}^{n}\left\{ n \atop k \right\} &= n!\sum_{k=0}^{n}\frac{k!}{n!}\left\{ n \atop k \right\}\frac{1}{k!} = \\ &= n![t^n]\left[e^y \;\middle|\; y = e^t - 1\right] = \\ &= n![t^n]\exp(e^t-1);\end{aligned}$$

therefore we have:

$$\mathcal{G}\left(\frac{\mathcal{B}_n}{n!}\right) = \exp(e^t-1).$$

We also defined the ordered Bell numbers as $\mathcal{O}_n = \sum_{k=0}^{n}\left\{ n \atop k \right\}k!$; therefore we have:

$$\begin{aligned}\frac{\mathcal{O}_n}{n!} &= \sum_{k=0}^{n}\frac{k!}{n!}\left\{ n \atop k \right\} = \\ &= [t^n]\left[\frac{1}{1-y} \;\middle|\; y = e^t - 1\right] = [t^n]\frac{1}{2-e^t}.\end{aligned}$$

We have thus obtained the exponential generating function:

$$\mathcal{G}\left(\frac{\mathcal{O}_n}{n!}\right) = \frac{1}{2-e^t}.$$

Stirling numbers of the two kinds are related between them in various ways. For example, we have:

$$\begin{aligned}\sum_k\begin{bmatrix} n \\ k \end{bmatrix}\left\{ k \atop m \right\} &= \frac{n!}{m!}\sum_k\frac{k!}{n!}\begin{bmatrix} n \\ k \end{bmatrix}\frac{m!}{k!}\left\{ k \atop m \right\} = \\ &= \frac{n!}{m!}[t^n]\left[(e^y-1)^m \;\middle|\; y = \ln\frac{1}{1-t}\right] = \\ &= \frac{n!}{m!}[t^n]\frac{t^m}{(1-t)^m} = \frac{n!}{m!}\binom{n-1}{m-1}.\end{aligned}$$

Besides, two orthogonality relations exist between Stirling numbers. The first one is proved in this way:

$$\sum_k\begin{bmatrix} n \\ k \end{bmatrix}\left\{ k \atop m \right\}(-1)^{n-k} =$$

$$
\begin{aligned}
&= (-1)^n \frac{n!}{m!} \sum_k \frac{k!}{n!} \begin{bmatrix} n \\ k \end{bmatrix} \frac{m!}{k!} \begin{Bmatrix} k \\ m \end{Bmatrix} (-1)^k = \\
&= (-1)^n \frac{n!}{m!} [t^n] \left[(e^{-y}-1)^m \;\middle|\; y = \ln \frac{1}{1-t} \right] = \\
&= (-1)^n \frac{n!}{m!} [t^n] (-t)^m = \delta_{n,m}.
\end{aligned}
$$

The second orthogonality relation is proved in a similar way and reads:

$$
\sum_k \begin{Bmatrix} n \\ k \end{Bmatrix} \begin{bmatrix} k \\ m \end{bmatrix} (-1)^{n-k} = \delta_{n,m}.
$$

We introduced Stirling numbers by means of Stirling identities relative to powers and falling factorials. We can now prove these identities by using a Riordan array approach. In fact:

$$
\begin{aligned}
&\sum_{k=0}^{n} \begin{bmatrix} n \\ k \end{bmatrix} (-1)^{n-k} x^k = \\
&= (-1)^n n! \sum_{k=0}^{n} \frac{k!}{n!} \begin{bmatrix} n \\ k \end{bmatrix} \frac{x^k}{k!} (-1)^k = \\
&= (-1)^n n! [t^n] \left[e^{-xy} \;\middle|\; y = \ln \frac{1}{1-t} \right] = \\
&= (-1)^n n! [t^n] (1-t)^x = n! \binom{x}{n} = x^{\underline{n}}
\end{aligned}
$$

and:

$$
\begin{aligned}
&\sum_{k=0}^{n} \begin{Bmatrix} n \\ k \end{Bmatrix} x^{\underline{k}} = n! \sum_{k=0}^{n} \frac{k!}{n!} \begin{Bmatrix} n \\ k \end{Bmatrix} \binom{x}{k} = \\
&= n! [t^n] \left[(1+y)^x \;\middle|\; y = e^t - 1 \right] = \\
&= n! [t^n] e^{tx} = x^n.
\end{aligned}
$$

We conclude this section by showing two possible connections between Stirling numbers and Bernoulli numbers. First we have:

$$
\begin{aligned}
&\sum_{k=0}^{n} \begin{Bmatrix} n \\ k \end{Bmatrix} \frac{(-1)^k k!}{k+1} = n! \sum_{k=0}^{n} \frac{k!}{n!} \begin{Bmatrix} n \\ k \end{Bmatrix} \frac{(-1)^k}{k+1} = \\
&= n! [t^n] \left[-\frac{1}{y} \ln \frac{1}{1+y} \;\middle|\; y = e^t - 1 \right] = \\
&= n! [t^n] \frac{t}{e^t - 1} = B_n
\end{aligned}
$$

which proves that Bernoulli numbers can be defined in terms of the Stirling numbers of the second kind. For the Stirling numbers of the first kind we have the identity:

$$
\begin{aligned}
\sum_{k=0}^{n} \begin{bmatrix} n \\ k \end{bmatrix} B_k &= n! \sum_{k=0}^{n} \frac{k!}{n!} \begin{bmatrix} n \\ k \end{bmatrix} \frac{B_k}{k!} = \\
&= n! [t^n] \left[\frac{y}{e^y - 1} \;\middle|\; y = \ln \frac{1}{1-t} \right] = \\
&= n! [t^n] \frac{1-t}{t} \ln \frac{1}{1-t} = \\
&= n! [t^{n+1}] (1-t) \ln \frac{1}{1-t} = \frac{(n-1)!}{n+1}.
\end{aligned}
$$

Clearly, this holds for $n > 0$. For $n = 0$ we have:

$$
\sum_{k=0}^{n} \begin{bmatrix} n \\ k \end{bmatrix} B_k = B_0 = 1.
$$

Chapter 6

Formal methods

6.1 Formal languages

During the 1950's, the linguist Noam Chomski introduced the concept of a *formal language*. Several definitions have to be provided before a precise statement of the concept can be given. Therefore, let us proceed in the following way.

First, we recall definitions given in Section 2.1. An *alphabet* is a finite set $A = \{a_1, a_2, \ldots, a_n\}$, whose elements are called *symbols* or *letters*. A *word* on A is a finite sequence of symbols in A; the sequence is written by juxtaposing the symbols, and therefore a word w is denoted by $w = a_{i_1} a_{i_2} \ldots a_{i_r}$, and $r = |w|$ is the *length* of the word. The empty sequence is called the *empty word* and is conventionally denoted by ϵ; its length is obviously 0, and is the only word of 0 length.

The set of all the words on A, the empty word included, is indicated by A^*, and by A^+ if the empty word is excluded. Algebraically, A^* is the *free monoid* on A, that is the monoid freely generated by the symbols in A. To understand this point, let us consider the operation of *juxtaposition* and recursively apply it starting with the symbols in A. What we get are the words on A, and the juxtaposition can be seen as an operation between them. The algebraic structure thus obtained has the following properties:

1. associativity: $w_1(w_2 w_3) = (w_1 w_2)w_3 = w_1 w_2 w_3$;
2. ϵ is the *identity* or *neutral element*: $\epsilon w = w\epsilon = w$.

It is called a *monoid*, which, by construction, has been generated by combining the symbols in A in all the possible ways. Because of that, $(A^*, \cdot)$, if $\cdot$ denotes the juxtaposition, is called the "free monoid" generated by A. Observe that a monoid is an algebraic structure more general than a group, in which all the elements have an inverse as well.

If $w \in A^*$ and z is a word such that w can be decomposed $w = w_1 z w_2$ (w_1 and/or w_2 possibly empty), we say that z is a *subword* of w; we also say that z *occurs* in w, and the particular instance of z in w is called an *occurrence* of z in w. Observe that if z is a subword of w, it can have more than one occurrence in w. If $w = z w_2$, we say that z is a *head* or *prefix* of w, and if $w = w_1 z$, we say that z is a *tail* or *suffix* of w. Finally, a *language* on A is any subset $L \subseteq A^*$.

The basic definition concerning formal languages is the following: a *grammar* is a 4-tuple $G = (T, N, \sigma, \mathcal{P})$, where:

- $T = \{a_1, a_2, \ldots, a_n\}$ is the alphabet of *terminal symbols*;
- $N = \{\phi_1, \phi_2, \ldots, \phi_m\}$ is the alphabet of *non-terminal symbols*;
- $\sigma \in N$ is the *initial symbol*;
- $\mathcal{P}$ is a finite set of *productions*.

Usually, the symbols in T are denoted by lower case Latin letters; the symbols in N by Greek letters or by upper case Latin letters. A *production* is a pair (z_1, z_2) of words in $T \cup N$, such that z_1 contains at least a symbol in N; the production is often indicated by $z_1 \to z_2$. If $w \in (T \cup N)^*$, we can apply a production $z_1 \to z_2 \in \mathcal{P}$ to w whenever w can be decomposed $w = w_1 z_1 w_2$, and the result is the new word $w_1 z_2 w_2 \in (T \cup N)^*$; we will write $w = w_1 z_1 w_2 \vdash w_1 z_2 w_2$ when $w_1 z_1 w_2$ is the decomposition of w in which z_1 is the *leftmost occurrence* of z_1 in w; in other words, if we also have $w = \widehat{w}_1 z_1 \widehat{w}_2$, then $|w_1| < |\widehat{w}_1|$.

Given a grammar $G = (T, N, \sigma, \mathcal{P})$, we define the relation $w \vdash \widehat{w}$ between words $w, \widehat{w} \in (T \cup N)^*$: the relation holds if and only if a production $z_1 \to z_2 \in \mathcal{P}$ exists such that z_1 occurs in w, $w = w_1 z_1 w_2$ is the leftmost occurrence of z_1 in w and $\widehat{w} = w_1 z_2 w_2$. We also denote by $\vdash^*$ the transitive closure of $\vdash$ and call it *generation* or *derivation*; this means that $w \vdash^* \widehat{w}$ if and only if a sequence $(w = w_1, w_2, \ldots, w_s = \widehat{w})$ exists such that $w_1 \vdash w_2, w_2 \vdash w_3, \ldots, w_{s-1} \vdash w_s$. We observe explicitly that by our condition that in every production $z_1 \to z_2$ the word z_1 should contain

at least a symbol in N, if a word $w_i \in T^*$ is produced during a generation, it is *terminal*, i.e., the generation should stop. By collecting all these definitions, we finally define the *language generated by the grammar* G as the set:

$$L(G) = \left\{ w \in T^* \;\middle|\; \sigma \vdash^* w \right\}$$

i.e., a word $w \in T^*$ is in $L(G)$ if and only if we can generate it by starting with the initial symbol σ and go on by applying the productions in $\mathcal{P}$ until w is generated. At that moment, the generation stops. Note that, sometimes, the generation can go on forever, never generating a word on T; however, this is not a problem: it only means that such generations should be ignored.

6.2 Context-free languages

The definition of a formal language is quite general and it is possible to show that formal languages coincide with the class of "partially recursive sets", the largest class of sets which can be constructed recursively, i.e., in finite terms. This means that we can give rules to build such sets (e.g., we can give a grammar for them), but their construction can go on forever, so that, looking at them from another point of view, if we wish to know whether a word w belongs to such a set S, we can be unlucky and an infinite process can be necessary to find out that $w \notin S$.

Because of that, people have studied more restricted classes of languages, for which a finite process is always possible for finding out whether w belongs to the language or not. Surely, the most important class of this kind is the class of "context-free languages". They are defined in the following way. A *context-free grammar* is a grammar $G = (T, N, \sigma, \mathcal{P})$ in which all the productions $z_1 \to z_2$ in $\mathcal{P}$ are such that $z_1 \in N$. The naming "context-free" derives from this definition, because a production $z_1 \to z_2$ is applied whenever the non-terminal symbol z_1 is the leftmost non-terminal symbol in a word, irrespective of the context in which it appears.

As a very simple example, let us consider the following grammar. Let $T = \{a, b\}$, $N = \{\sigma\}$; σ is the initial symbol of the grammar, being the only non-terminal. The set $\mathcal{P}$ is composed of the two productions:

$$\sigma \to \epsilon \qquad\qquad \sigma \to a\sigma b\sigma.$$

This grammar is called the *Dyck grammar* and the language generated by it the *Dyck language*. In Figure 6.1 we draw the generation of some words in the Dyck language. The recursive nature of the productions allows us to prove properties of the Dyck language by means of mathematical induction:

Theorem 6.2.1 *A word $w \in \{a, b\}^*$ belongs to the Dyck language D if and only if:*

i) the number of a's in w equals the number of b's;

ii) in every prefix z of w the number of a's is not less than the number of b's.

Proof: Let $w \in D$; if $w = \epsilon$ nothing has to be proved. Otherwise, w is generated by the second production and $w = aw_1bw_2$ with $w_1, w_2 \in D$; therefore, if we suppose that i) holds for w_1 and w_2, it also holds for w. For ii), any prefix z of w must have one of the forms: a, az_1 where z_1 is a prefix of w_1, aw_1b or aw_1bz_2 where z_2 is a prefix of w_2. By the induction hypothesis, ii) should hold for z_1 and z_2, and therefore it is easily proved for w. Vice versa, let us suppose that i) and ii) hold for $w \in T^*$. If $w \neq \epsilon$, then by ii) w should begin by a. Let us scan w until we find the first occurrence of the symbol b such that $w = aw_1bw_2$ and in w_1 the number of b's equals the number of a's. By i) such occurrence of b must exist, and consequently w_1 and w_2 must satisfy condition i). Besides, if w_1 and w_2 are not empty, then they should satisfy condition ii), by the very construction of w_1 and the fact that w satisfies condition ii) by hypothesis. We have thus obtained a decomposition of w showing that the second production has been used. This completes the proof. ∎

If we substitute the letter a with the symbol '(' and the letter b with the symbol ')', the theorem shows that the words in the Dyck language are the possible parenthetizations of an expression. Therefore, the number of Dyck words with n pairs of parentheses is the Catalan number $\binom{2n}{n}/(n+1)$. We will see how this result can also be obtained by starting with the definition of the Dyck language and applying a suitable and mechanical method, known as *Schützenberger methodology* or *symbolic method*. The method can be applied to every set of objects, which are defined through a non-ambiguous context-free grammar.

A context-free grammar G is *ambiguous* iff there exists a word $w \in L(G)$ which can be generated by two different leftmost derivations. In other words, a context-free grammar H is *non-ambiguous* iff every word $w \in L(H)$ can be generated in one and only one way. An example of an ambiguous grammar is $G = (T, N, \sigma, \mathcal{P})$ where $T = \{1\}$, $N = \{\sigma\}$ and $\mathcal{P}$ contains the two productions:

$$\sigma \to 1 \qquad\qquad \sigma \to \sigma\sigma.$$

For example, the word 111 is generated by the two following leftmost derivations:

$$\sigma \vdash \sigma\sigma \vdash 1\sigma \vdash 1\sigma\sigma \vdash 11\sigma \vdash 111$$

$$\sigma \vdash \sigma\sigma \vdash \sigma\sigma\sigma \vdash 1\sigma\sigma \vdash 11\sigma \vdash 111.$$

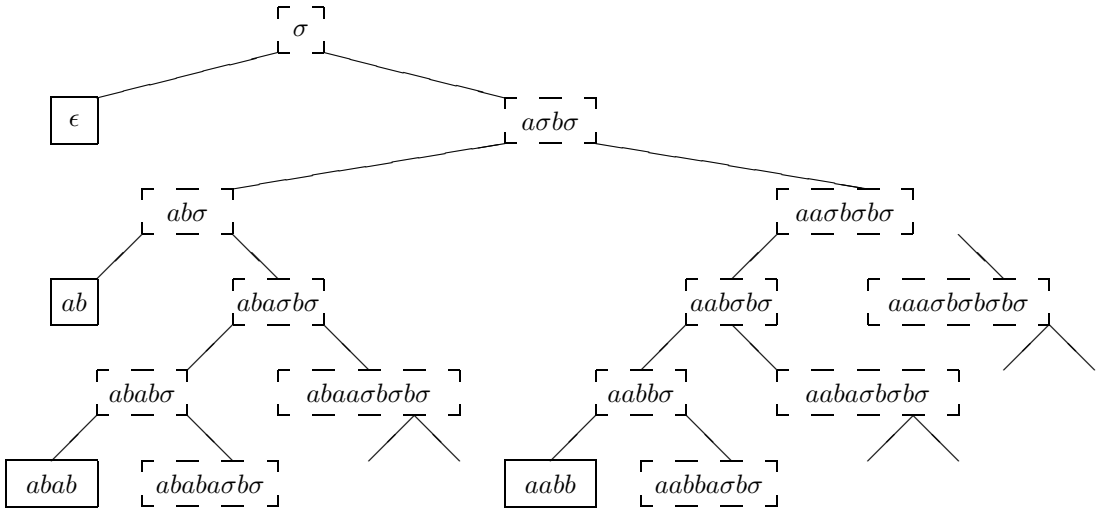


Figure 6.1: The generation of some Dyck words

Instead, the Dyck grammar is non-ambiguous; in fact, as we have shown in the proof of the previous theorem, given any word $w \in D$, $w \neq \epsilon$, there is only one decomposition $w = aw_1bw_2$, having $w_1, w_2 \in D$; therefore, w can only be generated in a single way. In general, if we show that any word in a context-free language $L(G)$, generated by some grammar G, has a unique decomposition according to the productions in G, then the grammar cannot be ambiguous. Because of the connection between the Schützenberger methodology and non-ambiguous context-free grammars, we are mainly interested in this kind of grammars. For the sake of completeness, a context-free language is called *intrinsically ambiguous* iff every context-free grammar generating it is ambiguous. This definition stresses the fact that, if a language is generated by an ambiguous grammar, it can also be generated by some non-ambiguous grammar, unless it is intrinsically ambiguous. It is possible to show that intrinsically ambiguous languages actually exist; fortunately, they are not very frequent. For example, the language generated by the previous ambiguous grammar is $\{1\}^+$, i.e., the set of all the words composed by any sequence of 1's, except the empty word. actually, it is not an ambiguous language and a non-ambiguous grammar generating it is given by the same T, N, σ and the two productions:

$$\sigma \to 1 \qquad \sigma \to 1\sigma.$$

It is a simple matter to show that every word $11\ldots1$ can be uniquely decomposed according to these productions.

6.3 Formal languages and programming languages

In 1960, the formal definition of the programming language ALGOL'60 was published. ALGOL'60 has surely been the most influential programming language ever created, although it was actually used only by a very limited number of programmers. Most of the concepts we now find in programming languages were introduced by ALGOL'60, of which, for example, PASCAL and C are direct derivations. Here, we are not interested in these aspects of ALGOL'60, but we wish to spend some words on how ALGOL'60 used context-free grammars to define its syntax in a formal and precise way. In practice, a program in ALGOL'60 is a word generated by a (rather complex) context-free grammar, whose initial symbol is ⟨`program`⟩.

The ALGOL'60 grammar used, as terminal symbol alphabet, the characters available on the standard keyboard of a computer; actually, they were the characters *punchable* on a card, the input mean used at that time to introduce a program into the computer. The non-terminal symbol notation was one of the most appealing inventions of ALGOL'60: the symbols were composed by entire English sentences enclosed by the two special parentheses ⟨ and ⟩. This allowed to clearly express the intended meaning of the non-terminal symbols. The previous example concerning ⟨`program`⟩ makes surely sense. Another technical device used by ALGOL'60 was the compaction of productions; if we had several production with the same left hand symbol $\beta \to w_1, \beta \to w_2, \ldots, \beta \to w_k$,

they were written as a single rule:

$$\beta ::= w_1 \,|\, w_2 \,|\, \cdots \,|\, w_k$$

where ::= was a metasymbol denoting definition and | was read "or" to denote alternatives. This notation is usually called *Backus Normal Form* (BNF).

Just to do a very simple example, in Figure 6.1 (lines 1 through 6) we show how integer numbers were defined. This definition avoids leading 0's in numbers, but allows both $+0$ and -0. Productions can be easily changed to avoid $+0$ or -0 or both. In the same figure, line 7 shows the definition of the conditional statements.

This kind of definition gives a precise formulation of all the clauses in the programming language. Besides, since the program has a single generation according to the grammar, it is possible to find this derivation starting from the actual program and therefore give its exact structure. This allows to give precise information to the compiler, which, in a sense, is directed from the formal syntax of the language (*syntax directed compilation*).

A very interesting aspect is how this context-free grammar definition can avoid ambiguities in the interpretation of a program. Let us consider an expression like $a + b * c$; according to the rules of Algebra, the multiplication should be executed before the addition, and the computer must follow this convention in order to create no confusion. This is done by the simplified productions given by lines 8 through 11 in Figure 6.1. The derivation of the simple expression $a+b*c$, or of a more complicated expression, reveals that it is decomposed into the sum of a and $b * c$; this information is passed to the compiler and the multiplication is actually performed before addition. If powers are also present, they are executed before products.

This ability of context-free grammars in designing the syntax of programming languages is very important, and after ALGOL'60 the syntax of every programming language has always been defined by context-free grammars. We conclude by remembering that a more sophisticated approach to the definition of programming languages was tried with ALGOL'68 by means of van Wijngaarden's grammars, but the method revealed too complex and was abandoned.

6.4 The symbolic method

The Schützenberger's method allows us to obtain the counting generating function for every non-ambiguous language, starting with the corresponding non-ambiguous grammar and proceeding in a mechanical way. Let us begin by a simple example; *Fibonacci words* are the words on the alphabet $\{0, 1\}$ beginning and ending by the symbol 1 and never containing two consecutive 0's. For small values of n, Fibonacci words of length n are easily displayed:

$n = 1$ 1
$n = 2$ 11
$n = 3$ 111, 101
$n = 4$ 1111, 1011, 1101
$n = 5$ 11111, 10111, 11011, 11101, 10101

If we count them by their length, we obtain the sequence $\{0, 1, 1, 2, 3, 5, 8, \ldots\}$, which is easily recognized as the Fibonacci sequence. In fact, a word of length n is obtained by adding a trailing 1 to a word of length $n-1$, or adding a trailing 01 to a word of length $n - 2$. This immediately shows, in a combinatorial way, that Fibonacci words are counted by Fibonacci numbers. Besides, we get the productions of a non-ambiguous context-free grammar $G = (T, N, \sigma, \mathcal{P})$, where $T = \{0, 1\}$, $N = \{\phi\}$, $\sigma = \phi$ and $\mathcal{P}$ contains:

$$\phi \to 1 \qquad \phi \to \phi 1 \qquad \phi \to \phi 01$$

(these productions could have been written $\phi ::= 1 \,|\, \phi 1 \,|\, \phi 01$ by using the ALGOL'60 notations).

We are now going to obtain the counting generating function for Fibonacci words by applying the Schützenberger's method. This consists in the following steps:

1. every non-terminal symbol $\sigma \in N$ is transformed into the name of its counting generating function $\sigma(t)$;
2. every terminal symbol is transformed into t;
3. the empty word is transformed into 1;
4. every | sign is transformed into a + sign, and ::= is transformed into an equal sign.

After having performed these transformations, we obtain a system of equations, which can be solved in the unknown generating functions introduced in the first step. They are the counting generating functions for the languages generated by the corresponding non-terminal symbols, when we consider them as the initial symbols.

The definition of the Fibonacci words produces:

$$\phi(t) = t + t\phi(t) + t^2\phi(t)$$

the solution of which is:

$$\phi(t) = \frac{t}{1 - t - t^2};$$

this is obviously the generating function for the Fibonacci numbers. Therefore, we have shown that the

1 ⟨digit⟩ ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
2 ⟨non − zero digit⟩ ::= 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
3 ⟨sequence of digits⟩ ::= ⟨digit⟩ | ⟨digit⟩ ⟨sequence of digits⟩
4 ⟨unsigned number⟩ ::= ⟨digit⟩ | ⟨non − zero digit⟩ ⟨sequence of digits⟩
5 ⟨signed number⟩ ::= +⟨unsigned number⟩ | − ⟨unsigned number⟩
6 ⟨integer number⟩ ::= ⟨unsigned number⟩ | ⟨signed number⟩

7 ⟨conditional clause⟩ ::= if ⟨condition⟩ then ⟨instruction⟩ |
if ⟨condition⟩ then ⟨instruction⟩ else ⟨instruction⟩

8 ⟨expression⟩ ::= ⟨term⟩ | ⟨term⟩ + ⟨expression⟩
9 ⟨term⟩ ::= ⟨factor⟩ | ⟨term⟩ ∗ ⟨factor⟩
10 ⟨factor⟩ ::= ⟨element⟩ | ⟨factor⟩ ↑ ⟨element⟩
11 ⟨element⟩ ::= ⟨constant⟩ | ⟨variable⟩ | (⟨expression⟩)

Table 6.1: Context-free languages and programming languages

number of Fibonacci words of length n is F_n, as we have already proved by combinatorial arguments.

In the case of the Dyck language, the definition yields:

$$\sigma(t) = 1 + t^2\sigma(t)^2$$

and therefore:

$$\sigma(t) = \frac{1 - \sqrt{1 - 4t^2}}{2t^2}.$$

Since every word in the Dyck language has an even length, the number of Dyck words with $2n$ symbols is just the nth Catalan number, and this also we knew by combinatorial means.

Another example is given by the *Motzkin words*; these are words on the alphabet $\{a, b, c\}$ in which a, b act as parentheses in the Dyck language, while c is free and can appear everywhere. Therefore, the definition of the language is:

$$\mu ::= \epsilon \,|\, c\mu \,|\, a\mu b\mu$$

if μ is the only non-terminal symbol. The Schützenberger's method gives the equation:

$$\mu(t) = 1 + t\mu(t) + t^2\mu(t)^2$$

whose solution is easily found:

$$\mu(t) = \frac{1 - t - \sqrt{1 - 2t - 3t^2}}{2t^2}.$$

By expanding this function we find the sequence of *Motzkin numbers*, beginning:

n	0	1	2	3	4	5	6	7	8	9
M_n	1	1	2	4	9	21	51	127	323	835

These numbers count the so-called *unary-binary trees*, i.e., trees the nodes of which have ariety 1 or 2. They can be defined in a pictorial way by means of an *object grammar*:

An object grammar defines combinatorial objects instead of simple letters or words; however, most times, it is rather easy to pass from an object grammar to an equivalent context-free grammar, and therefore obtain counting generating functions by means of Schützenberger's method. For example, the object grammar in Figure 6.2 is obviously equivalent to the context-free grammar for Motzkin words.

6.5 The bivariate case

In Schützenberger's method, the rôle of the indeterminate t is to count the number of letters or symbols occurring in the generated words; because of that, every symbol appearing in a production is transformed into a t. However, we can wish to count other parameters instead of or in conjunction with the number of symbols. This is accomplished by modifying the intended meaning of the indeterminate t and/or introducing some other indeterminate to take into account the other parameters.

For example, in the Dyck language, we can wish to count the number of pairs a, b occurring in the words; this means that t no longer counts the single letters, but counts the pairs. Therefore, the Schützenberger's method gives the equation $\sigma(t) = 1 + t\sigma(t)^2$, whose solution is just the generating function of the Catalan numbers.

An interesting application is as follows. Let us suppose we wish to know how many Fibonacci words of length n contain k zeroes. Besides the indeterminate t counting the total number of symbols, we introduce a new indeterminate z counting the number of zeroes. From the productions of the Fibonacci grammar, we derive an equation for the bivariate generating function $\phi(t, z)$, in which the coefficient of $t^n z^k$ is just the

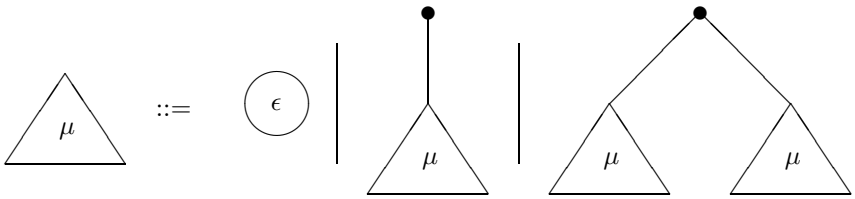


Figure 6.2: An object grammar

number we are looking for. The equation is:

$$\phi(t,z) = t + t\phi(t,z) + t^2 z\phi(t,z).$$

In fact, the production $\phi \to \phi 1$ increases by 1 the length of the words, but does not introduce any 0; the production $\phi \to \phi 01$, increases by 2 the length and introduces a zero. By solving this equation, we find:

$$\phi(t,z) = \frac{t}{1-t-zt^2}.$$

We can now extract the coefficient $\phi_{n,k}$ of $t^n z^k$ in the following way:

$$\begin{aligned}
\phi_{n,k} &= [t^n][z^k]\frac{t}{1-t-zt^2} = \\
&= [t^n][z^k]\frac{t}{1-t}\frac{1}{1-z\frac{t^2}{1-t}} = \\
&= [t^n][z^k]\frac{t}{1-t}\sum_{k=0}^{\infty} z^k\left(\frac{t^2}{1-t}\right)^k = \\
&= [t^n][z^k]\sum_{k=0}^{\infty}\frac{t^{2k+1}z^k}{(1-t)^{k+1}} = [t^n]\frac{t^{2k+1}}{(1-t)^{k+1}} = \\
&= [t^{n-2k-1}]\frac{1}{(1-t)^{k+1}} = \binom{n-k-1}{k}.
\end{aligned}$$

Therefore, the number of Fibonacci words of length n containing k zeroes is counted by a binomial coefficient. The second expression in the derivation shows that the array $(\phi_{n,k})_{n,k\in N}$ is indeed a Riordan array $(t/(1-t), t/(1-t))$, which is the Pascal triangle stretched vertically, i.e., column k is shifted down by k positions ($k+1$, in reality). The general formula we know for the row sums of a Riordan array gives:

$$\begin{aligned}
\sum_k \phi_{n,k} &= \sum_k \binom{n-k-1}{k} = \\
&= [t^n]\frac{t}{1-t}\left[\frac{1}{1-y}\;\middle|\; y = \frac{t^2}{1-t}\right] = \\
&= [t^n]\frac{t}{1-t-t^2} = F_n
\end{aligned}$$

as we were expecting. A more interesting problem is to find the average number of zeroes in all the Fibonacci words with n letters. First, we count the total number of zeroes in all the words of length n:

$$\begin{aligned}
\sum_k k\phi_{n,k} &= \sum_k \binom{n-k-1}{k}k = \\
&= [t^n]\frac{t}{1-t}\left[\frac{y}{(1-y)^2}\;\middle|\; y = \frac{t^2}{1-t}\right] = \\
&= [t^n]\frac{t^3}{(1-t-t^2)^2}.
\end{aligned}$$

We extract the coefficient:

$$\begin{aligned}
&[t^n]\frac{t^3}{(1-t-t^2)^2} = \\
&= [t^{n-1}]\left(\frac{1}{\sqrt{5}}\left(\frac{1}{1-\phi t} - \frac{1}{1-\widehat{\phi}t}\right)\right)^2 = \\
&= \frac{1}{5}[t^{n-1}]\frac{1}{(1-\phi t)^2} - \frac{2}{5}[t^n]\frac{t}{(1-\phi t)(1-\widehat{\phi}t)} + \\
&\quad + \frac{1}{5}[t^{n-1}]\frac{1}{(1-\widehat{\phi}t)^2} = \\
&= \frac{1}{5}[t^{n-1}]\frac{1}{(1-\phi t)^2} - \frac{2}{5\sqrt{5}}[t^n]\frac{1}{1-\phi t} + \\
&\quad + \frac{2}{5\sqrt{5}}[t^n]\frac{1}{1-\widehat{\phi}t} + \frac{1}{5}[t^{n-1}]\frac{1}{(1-\widehat{\phi}t)^2}.
\end{aligned}$$

The last two terms are negligible because they rapidly tend to 0; therefore we have:

$$\sum_k k\phi_{n,k} \approx \frac{n}{5}\phi^{n-1} - \frac{2}{5\sqrt{5}}\phi^n.$$

To obtain the average number Z_n of zeroes, we need to divide this quantity by $F_n \sim \phi^n/\sqrt{5}$, the total number of Fibonacci words of length n:

$$Z_n = \frac{\sum_k k\phi_{n,k}}{F_n} \sim \frac{n}{\phi\sqrt{5}} - \frac{2}{5} = \frac{5-\sqrt{5}}{10}n - \frac{2}{5}.$$

This shows that the average number of zeroes grows linearly with the length of the words and tends to become the 27.64% of this length, because $(5-\sqrt{5})/10 \approx 0.2763932022\ldots$.

6.6 The Shift Operator

In the usual mathematical terminology, an *operator* is a mapping from some set $\mathcal{F}_1$ of functions into some

other set of functions $\mathcal{F}_2$. We have already encountered the operator $\mathcal{G}$, acting from the set of sequences (which are properly functions from $\mathbb{N}$ into $\mathbb{R}$ or $\mathbb{C}$) into the set of formal power series (analytic functions). Other usual examples are D, the operator of differentiation, and $\int$, the operator of indefinite integration. Since the middle of the 19th century, the English mathematicians (G. Boole in particular) introduced the concept of a *finite operator*, i.e., an operator acting in finite terms, in contrast to an *infinitesimal operator*, such as differentiation and integration. The most simple among the finite operators is the *identity*, denoted by I or by 1, which does not change anything. Operationally, however, the most important finite operator is the *shift operator*, denoted by E, changing the value of any function f at a point x into the value of the same function at the point $x+1$. We write:

$$Ef(x) = f(x+1)$$

Unlike an infinitesimal operator as D, a finite operator can be applied to a sequence, as well, and in that case we can write:

$$Ef_n = f_{n+1}$$

This property is particularly interesting from our point of view but, in order to follow the usual notational conventions, we shall adopt the first, functional notation rather than the second one, more specific for sequences.

The shift operator can be iterated and, to denote the successive applications of E, we write E^n instead of $EE\ldots E$ (n times). So:

$$E^2 f(x) = EEf(x) = Ef(x+1) = f(x+2)$$

and in general:

$$E^n f(x) = f(x+n)$$

Conventionally, we set $E^0 = I = 1$, and this is in accordance with the meaning of I:

$$E^0 f(x) = f(x) = If(x)$$

We wish to observe that every recurrence can be written using the shift operator. For example, the recurrence for the Fibonacci numbers is:

$$E^2 F_n = EF_n + F_n$$

and this can be written in the following way, separating the operator parts from the sequence parts:

$$(E^2 - E - I)F_n = 0$$

Some obvious properties of the shift operator are:

$$E(\alpha f(x) + \beta g(x)) = \alpha Ef(x) + \beta Eg(x)$$

$$E(f(x)g(x)) = Ef(x)Eg(x)$$

Hence, if c is any constant (i.e., $c \in \mathbb{R}$ or $c \in \mathbb{C}$) we have $Ec = c$.

It is possible to consider negative powers of E, as well. So we have:

$$E^{-1}f(x) = f(x-1) \qquad E^{-n}f(x) = f(x-n)$$

and this is in accordance with the usual rules of powers:

$$E^n E^m = E^{n+m} \qquad \text{for} \qquad n, m \in \mathbb{Z}$$

$$E^n E^{-n} = E^0 = I \qquad \text{for} \qquad n \in \mathbb{Z}$$

Finite operators are commonly used in Numerical Analysis. In that case, an increment h is defined and the shift operator acts according to this increment, i.e., $Ef(x) = f(x+h)$. When considering sequences, this makes no sense and we constantly use $h = 1$.

We can have occasions to use two or more shift operators, that is shift operators related to different variables. We'll distinguish them by suitable subscripts:

$$E_x f(x,y) = f(x+1,y) \qquad E_y f(x,y) = f(x,y+1)$$

$$E_x E_y f(x,y) = E_y E_x f(x,y) = f(x+1,y+1)$$

6.7 The Difference Operator

The second most important finite operator is the *difference operator* Δ; it is defined in terms of the shift and identity operators:

$$\begin{aligned} \Delta f(x) &= (E-1)f(x) = \\ &= Ef(x) - If(x) = f(x+1) - f(x) \end{aligned}$$

Some simple examples are:

$$\begin{aligned} \Delta c &= Ec - c = c - c = 0 \qquad \forall c \in \mathbb{C} \\ \Delta x &= Ex - x = x + 1 - x = 1 \\ \Delta x^2 &= Ex^2 - x^2 = (x+1)^2 - x^2 = 2x + 1 \\ \Delta x^n &= Ex^n - x^n = (x+1)^n - x^n = \\ &\quad \sum_{k=0}^{n-1} \binom{n}{k} x^k \\ \Delta \frac{1}{x} &= \frac{1}{x+1} - \frac{1}{x} = -\frac{1}{x(x+1)} \\ \Delta \binom{x}{m} &= \binom{x+1}{m} - \binom{x}{m} = \binom{x}{m-1} \end{aligned}$$

The last example makes use of the recurrence relation for binomial coefficients. An important observation

concerns the behavior of the difference operator with respect to the falling factorial:

$$\begin{aligned}\Delta x^{\underline{m}} &= (x+1)^{\underline{m}} - x^{\underline{m}} = (x+1)x(x-1)\cdots \\ &\quad \cdots(x-m+2) - x(x-1)\cdots(x-m+1) = \\ &= x(x-1)\cdots(x-m+2)(x+1-x+m-1) = \\ &= m x^{\underline{m-1}}\end{aligned}$$

This is analogous to the usual rule for the differentiation operator applied to x^m:

$$Dx^m = mx^{m-1}$$

As we shall see, many formal properties of the difference operator are similar to the properties of the differentiation operator. The rôle of the powers x^m is however taken by the falling factorials, which therefore assume a central position in the theory of finite operators.

The following general rules are rather obvious:

$$\Delta(\alpha f(x) + \beta g(x)) = \alpha \Delta f(x) + \beta \Delta g(x)$$

$$\begin{aligned}&\Delta(f(x)g(x)) = \\ &= E(f(x)g(x)) - f(x)g(x) = \\ &= f(x+1)g(x+1) - f(x)g(x) = \\ &= f(x+1)g(x+1) - f(x)g(x+1) + \\ &\quad + f(x)g(x+1) - f(x)g(x) = \\ &= \Delta f(x) E g(x) + f(x)\Delta g(x)\end{aligned}$$

resembling the differentiation rule $D(f(x)g(x)) = (Df(x))g(x) + f(x)Dg(x)$. In a similar way we have:

$$\begin{aligned}\Delta\frac{f(x)}{g(x)} &= \frac{f(x+1)}{g(x+1)} - \frac{f(x)}{g(x)} = \\ &= \frac{f(x+1)g(x) - f(x)g(x)}{g(x)g(x+1)} + \\ &\quad + \frac{f(x)g(x) - f(x)g(x+1)}{g(x)g(x+1)} = \\ &= \frac{(\Delta f(x))g(x) - f(x)\Delta g(x)}{g(x)Eg(x)}\end{aligned}$$

The difference operator can be iterated:

$$\begin{aligned}\Delta^2 f(x) &= \Delta\Delta f(x) = \Delta(f(x+1) - f(x)) = \\ &= f(x+2) - 2f(x+1) + f(x).\end{aligned}$$

From a formal point of view, we have:

$$\Delta^2 = (E-1)^2 = E^2 - 2E + 1$$

and in general:

$$\begin{aligned}\Delta^n &= (E-1)^n = \sum_{k=0}^{n}\binom{n}{k}(-1)^{n-k}E^k = \\ &= (-1)^n \sum_{k=0}^{n}\binom{n}{k}(-E)^k\end{aligned}$$

This is a very important formula, and it is the first example for the interest of combinatorics and generating functions in the theory of finite operators. In fact, let us iterate Δ on $f(x) = 1/x$:

$$\begin{aligned}\Delta^2\frac{1}{x} &= \frac{-1}{(x+1)(x+2)} + \frac{1}{x(x+1)} = \\ &= \frac{-x+x+2}{x(x+1)(x+2)} = \frac{2}{x(x+1)(x+2)} \\ \Delta^n\frac{1}{x} &= \frac{(-1)^n n!}{x(x+1)\cdots(x+n)}\end{aligned}$$

as we can easily show by mathematical induction. In fact:

$$\begin{aligned}\Delta^{n+1}\frac{1}{x} &= \frac{(-1)^n n!}{(x+1)\cdots(x+n+1)} - \\ &\quad - \frac{(-1)^n n!}{x(x+1)\cdots(x+n)} = \\ &= \frac{(-1)^{n+1}(n+1)!}{x(x+1)\cdots(x+n+1)}\end{aligned}$$

The formula for Δ^n now gives the following identity:

$$\Delta^n\frac{1}{x} = \sum_{k=0}^{n}\binom{n}{k}(-1)^{n-k}E^k\frac{1}{x}$$

By multiplying everything by $(-1)^n$ this identity can be written as:

$$\frac{n!}{x(x+1)\cdots(x+n)} = \frac{1}{x\binom{x+n}{n}} = \sum_{k=0}^{n}\binom{n}{k}\frac{(-1)^k}{x+k}$$

and therefore we have both a way to express the inverse of a binomial coefficient as a sum and an expression for the partial fraction expansion of the polynomial $x(x+1)\cdots(x+n)$ inverse.

6.8 Shift and Difference Operators - Example I

As the difference operator Δ can be expressed in terms of the shift operator E, so E can be expressed in terms of Δ:

$$E = \Delta + 1$$

This rule can be iterated, giving the summation formula:

$$E^n = (\Delta+1)^n = \sum_{k=0}^{n}\binom{n}{k}\Delta^k$$

which can be seen as the "dual" formula of the one already considered:

$$\Delta^n = \sum_{k=0}^{n}\binom{n}{k}(-1)^{n-k}E^k$$

The evaluation of the successive differences of any function $f(x)$ allows us to state and prove two identities, which may have combinatorial significance. Here we record some typical examples; we mark with an asterisk the cases when $\Delta^0 f(x) \neq If(x)$.

1) The function $f(x) = 1/x$ has already been developed, at least partially:

$$\begin{aligned}\Delta\frac{1}{x} &= \frac{-1}{x(x+1)}\\ \Delta^n\frac{1}{x} &= \frac{(-1)^n n!}{x(x+1)\cdots(x+n)}\frac{(-1)^n}{x}\binom{x+n}{n}^{-1}\end{aligned}$$

$$\sum_k \binom{n}{k}\frac{(-1)^k}{x+k} = \frac{n!}{x(x+1)\cdots(x+n)} =$$

$$= \frac{1}{x}\binom{x+n}{n}^{-1}$$

$$\sum_k \binom{n}{k}(-1)^k\binom{x+k}{k}^{-1} = \frac{x}{x+n}.$$

2*) A somewhat similar situation, but a bit more complex:

$$\begin{aligned}\Delta\frac{p+x}{m+x} &= \frac{m-p}{(m+x)(m+x+1)}\\ \Delta^n\frac{p+x}{m+x} &= \frac{m-p}{m+x}(-1)^{n-1}\binom{m+x+n}{n}^{-1}\end{aligned}$$

$$\sum_k \binom{n}{k}(-1)^k\frac{p+k}{m+k} = \frac{m-p}{m}\binom{m+n}{n}^{-1}\quad(n>0)$$

$$\sum_k \binom{n}{k}\binom{m+k}{k}^{-1}(-1)^k = \frac{m}{m+n}\quad\text{(see above).}$$

3) Another version of the first example:

$$\Delta\frac{1}{px+m} = \frac{-p}{(px+m)(px+p+m)}$$

$$\Delta^n\frac{1}{px+m} =$$

$$= \frac{(-1)^n n!p^n}{(px+m)(px+p+m)\cdots(px+np+m)}$$

According to this rule, we should have: $\Delta^0(p+x)/(m+x) = (p-m)(m+x)$; in the second next sum, however, we have to set $\Delta^0 = I$, and therefore we also have to subtract 1 from both members in order to obtain a true identity; a similar situation arises whenever we have $\Delta^0 \neq I$.

$$\sum_k \binom{n}{k}\frac{(-1)^k}{pk+m} = \frac{n!p^n}{m(m+p)\cdots(m+np)}$$

$$\sum_k \binom{n}{k}\frac{(-1)^k k!p^k}{m(m+p)\cdots(m+pk)} = \frac{1}{pn+m}$$

4*) A case involving the harmonic numbers:

$$\begin{aligned}\Delta H_x &= \frac{1}{x+1}\\ \Delta^n H_x &= \frac{(-1)^{n-1}}{n}\binom{x+n}{n}^{-1}\end{aligned}$$

$$\sum_k \binom{n}{k}(-1)^k H_{x+k} = -\frac{1}{n}\binom{x+n}{n}^{-1}\quad(n>0)$$

$$\sum_{k=1}^{n}\binom{n}{k}\frac{(-1)^{k-1}}{k}\binom{x+k}{k}^{-1} = H_{x+n} - H_x$$

where is to be noted the case $x = 0$.

5*) A more complicated case with the harmonic numbers:

$$\begin{aligned}\Delta x H_x &= H_x + 1\\ \Delta^n x H_x &= \frac{(-1)^n}{n-1}\binom{x+n-1}{n-1}^{-1}\end{aligned}$$

$$\sum_k \binom{n}{k}(-1)^k(x+k)H_{x+k} =$$

$$= \frac{1}{n-1}\binom{x+n-1}{n-1}^{-1}$$

$$\sum_k \binom{n}{k}\frac{(-1)^k}{k-1}\binom{x+k-1}{k-1}^{-1} =$$

$$= (x+n)\left(H_{x+n} - H_x\right) - n$$

6) Harmonic numbers and binomial coefficients:

$$\begin{aligned}\Delta\binom{x}{m}H_x &= \binom{x}{m-1}\left(H_x + \frac{1}{m}\right)\\ \Delta^n\binom{x}{m}H_x &= \binom{x}{m-n}\left(H_x + H_m - H_{m-n}\right)\end{aligned}$$

$$\sum_k \binom{n}{k}(-1)^k\binom{x+k}{m}H_{x+k} =$$

$$= (-1)^n\binom{x}{m-n}\left(H_x + H_m - H_{m-n}\right)$$

$$\sum_k \binom{n}{k}\binom{x}{m-k}\left(H_x + H_m - H_{m-k}\right) =$$

$$= \binom{x+n}{m}H_{x+n}$$

and by performing the sums on the left containing H_x and H_m:

$$\sum_k \binom{n}{k}\binom{x}{m-k}H_{m-k} =$$

$$= \binom{x+n}{m}\left(H_x + H_m - H_{x+n}\right)$$

7) The function $\ln(x)$ can be inserted in this group:

$$\begin{aligned}\Delta \ln(x) &= \ln\left(\frac{x+1}{x}\right)\\ \Delta^n \ln(x) &= (-1)^n \sum_k \binom{n}{k}(-1)^k \ln(x+k)\end{aligned}$$

$$\sum_k \binom{n}{k}(-1)^k \ln(x+k) =$$

$$= \sum_k \binom{n}{k}(-1)^k \ln(x+k)$$

$$\sum_{k=0}^{n}\sum_{j=0}^{k}(-1)^{k+j}\binom{n}{k}\binom{k}{j}\ln(x+j) = \ln(x+n)$$

Note that the last but one relation is an identity.

6.9 Shift and Difference Operators - Example II

Here we propose other examples of combinatorial sums obtained by iterating the shift and difference operators.

1) The typical sum containing the binomial coefficients: Newton's binomial theorem:

$$\begin{aligned}\Delta p^x &= (p-1)p^x\\ \Delta^n p^x &= (p-1)^n p^x\end{aligned}$$

$$\begin{aligned}\sum_k \binom{n}{k}(-1)^k p^k &= (p-1)^n\\ \sum_k \binom{n}{k}(p-1)^k &= p^n\end{aligned}$$

2) Two sums involving Fibonacci numbers:

$$\begin{aligned}\Delta F_x &= F_{x-1}\\ \Delta^n F_x &= F_{x-n}\end{aligned}$$

$$\begin{aligned}\sum_k \binom{n}{k}(-1)^k F_{x+k} &= (-1)^n F_{x-n}\\ \sum_k \binom{n}{k} F_{x-k} &= F_{x+n}\end{aligned}$$

3) Falling factorials are an introduction to binomial coefficients:

$$\begin{aligned}\Delta x^{\underline{m}} &= m x^{\underline{m-1}}\\ \Delta^n x^{\underline{m}} &= m^{\underline{n}} x^{\underline{m-n}}\end{aligned}$$

$$\begin{aligned}\sum_k \binom{n}{k}(-1)^k (x+k)^{\underline{m}} &= (-1)^n m^{\underline{n}} x^{\underline{m-n}}\\ \sum_k \binom{n}{k} m^{\underline{k}} x^{\underline{m-k}} &= (x+n)^{\underline{m}}\end{aligned}$$

4) Similar sums hold for raising factorials:

$$\begin{aligned}\Delta x^{\overline{m}} &= m(x+1)^{\overline{m-1}}\\ \Delta^n x^{\overline{m}} &= m^{\overline{n}}(x+n)^{\overline{m-n}}\end{aligned}$$

$$\begin{aligned}\sum_k \binom{n}{k}(-1)^k (x+k)^{\overline{m}} &= (-1)^n m^{\overline{n}}(x+n)^{\overline{m-n}}\\ \sum_k \binom{n}{k} m^{\overline{k}}(x+k)^{\overline{m-k}} &= (x+n)^{\overline{m}}\end{aligned}$$

5) Two sums involving the binomial coefficients:

$$\begin{aligned}\Delta\binom{x}{m} &= \binom{x}{m-1}\\ \Delta^n\binom{x}{m} &= \binom{x}{m-n}\end{aligned}$$

$$\begin{aligned}\sum_k \binom{n}{k}(-1)^k\binom{x+k}{m} &= (-1)^n\binom{x}{m-n}\\ \sum_k \binom{n}{k}\binom{x}{m-k} &= \binom{x+n}{m}\end{aligned}$$

6) Another case with binomial coefficients:

$$\begin{aligned}\Delta\binom{p+x}{m+x} &= \binom{p+x}{m+x+1}\\ \Delta^n\binom{p+x}{m+x} &= \binom{p+x}{m+x+n}\end{aligned}$$

$$\begin{aligned}\sum_k \binom{n}{k}(-1)^k\binom{p+k}{m+k} &= (-1)^n\binom{p}{m+n}\\ \sum_k \binom{n}{k}\binom{p}{m+k} &= \binom{p+n}{m+n}\end{aligned}$$

7) And now a case with the inverse of a binomial coefficient:

$$\begin{aligned}\Delta\binom{x}{m}^{-1} &= -\frac{m}{m-1}\binom{x+1}{m+1}^{-1}\\ \Delta^n\binom{x}{m}^{-1} &= (-1)^n\frac{m}{m+n}\binom{x+n}{m+n}^{-1}\end{aligned}$$

$$\sum_k \binom{n}{k}(-1)^k\binom{x+k}{m}^{-1} = \frac{m}{m+n}\binom{x+n}{m+n}^{-1}$$

$$\sum_k \binom{n}{k}(-1)^k\frac{m}{m+k}\binom{x+k}{m+k}^{-1} = \binom{x+n}{m}^{-1}$$

8) Two sums with the central binomial coefficients:

$$\Delta\frac{1}{4^x}\binom{2x}{x} = \frac{1}{2(x+1)}\frac{1}{4^x}\binom{2x}{x}$$
$$\Delta^n\frac{1}{4^x}\binom{2x}{x} = \frac{(-1)^n(2n)!}{n!(x+1)\cdots(x+n)}\frac{1}{4^n}\binom{2n}{n}$$

$$\sum_k\binom{n}{k}(-1)^k\binom{2x+2k}{x+k}\frac{1}{4^k} =$$
$$= \frac{(2n)!}{n!(x+1)\cdots(x+n)}\frac{1}{4^n}\binom{2x}{x} =$$
$$= \frac{1}{4^n}\binom{2n}{n}\binom{x+n}{n}^{-1}\binom{2x}{x}$$
$$\sum_k\binom{n}{k}\frac{(-1)^k(2k)!}{k!(x+1)\cdots(x+k)}\frac{1}{4^k}\binom{2x}{x} =$$
$$= \frac{1}{4^n}\binom{2x+2n}{x+n}$$

9) Two sums with the inverse of the central binomial coefficients:

$$\Delta 4^x\binom{2x}{x}^{-1} = \frac{4^x}{2x+1}\binom{2x}{x}^{-1}$$
$$\Delta^n 4^x\binom{2x}{x}^{-1} =$$
$$= \frac{1}{2n-1}\frac{(-1)^{n-1}(2n)!4^x}{2^n n!(2x+1)\cdots(2x+2n-1)}\binom{2x}{x}^{-1}$$

$$\sum_k\binom{n}{k}(-1)^k 4^k\binom{2x+2k}{x+k}^{-1} =$$
$$= \frac{1}{2n-1}\frac{(2n)!}{2^n n!(2x+1)\cdots(2x+2n-1)}\binom{2x}{x}^{-1}$$
$$\sum_k\binom{n}{k}\frac{1}{2k-1}\frac{(2k)!(-1)^{k-1}}{2^k k!(2x+1)\cdots(2x+2k-1)} =$$
$$= 4^n\binom{2x+2n}{x+n}^{-1}\binom{2x}{x} - 1$$

6.10 The Addition Operator

The *addition operator* S is analogous to the difference operator:

$$S = E + 1$$

and in fact a simple connection exists between the two operators:

$$S(-1)^x f(x) = (-1)^{x+1}f(x+1) + (-1)^x f(x) =$$
$$= (-1)^{x+1}(f(x+1) - f(x)) =$$
$$= (-1)^{x-1}\Delta f(x)$$

Because of this connection, the addition operator has not widely considered in the literature, and the symbol S only is used here for convenience. Likewise the difference operator, the addition operator can be iterated and often produces interesting combinatorial sums according to the rules:

$$S^n = (E+1)^n = \sum_k\binom{n}{k}E^k$$
$$E^n = (S-1)^n = \sum_k\binom{n}{k}(-1)^{n-k}S^k$$

Some examples are in order here:

1) Fibonacci numbers are typical:

$$SF_m = F_{m+1} + F_m = F_{m+2}$$
$$S^n F_m = F_{m+2n}$$

$$\sum_k\binom{n}{k}F_{m+k} = F_{m+2n}$$
$$\sum_k\binom{n}{k}(-1)^k F_{m+2k} = (-1)^n F_{m+n}$$

2) Here are the binomial coefficients:

$$S\binom{m}{x} = \binom{m+1}{x+1}$$
$$S^n\binom{m}{x} = \binom{m+n}{x+n}$$

$$\sum_k\binom{n}{k}\binom{m}{x+k} = \binom{m+n}{x+n}$$
$$\sum_k\binom{n}{k}(-1)^k\binom{m+k}{x+k} = (-1)^n\binom{m}{x+n}$$

3) And finally the inverse of binomial coefficients:

$$S\binom{m}{x}^{-1} = \frac{m+1}{m}\binom{m-1}{x}^{-1}$$
$$S^n\binom{m}{x} = \frac{m+1}{m-n+1}\binom{m-n}{x}^{-1}$$

$$\sum_k\binom{n}{k}\binom{m}{x+k}^{-1} = \frac{m+1}{m-n+1}\binom{m+n}{x}^{-1}$$

$$\sum_k\binom{n}{k}\frac{(-1)^k}{m-k+1}\binom{m+k}{x}^{-1} = \frac{1}{m+1}\binom{m}{x+n}^{-1}.$$

We can obviously invent as many expressions as we desire and, correspondingly, may obtain some summation formulas of combinatorial interest. For example:

$$S\Delta = (E+1)(E-1) = E^2 - 1 =$$
$$= (E-1)(E+1) = \Delta S$$

This derivation shows that the two operators S and Δ commute. We can directly verify this property:

$$\begin{aligned} S\Delta f(x) &= S(f(x+1)-f(x)) = \\ &= f(x+2)-f(x) = (E^2-1)f(x) \\ \Delta S f(x) &= \Delta(f(x+1)+f(x)) = \\ &= f(x+2)-f(x) = (E^2-1)f(x) \end{aligned}$$

Consequently, we have the two summation formulas:

$$\begin{aligned} \Delta^n S^n &= (E^2-1)^n = \sum_k \binom{n}{k}(-1)^{n-k}E^{2k} \\ E^{2n} &= (\Delta S+1)^n = \sum_k \binom{n}{k}\Delta^k S^k \end{aligned}$$

A simple example is offered by the Fibonacci numbers:

$$\begin{aligned} \Delta S F_m &= F_{m+1} \\ (\Delta S)^n F_m &= \Delta^n S^n F_m = F_{m+n} \end{aligned}$$

$$\begin{aligned} \sum_k \binom{n}{k}(-1)^k F_{m+2k} &= (-1)^n F_{m+n} \\ \sum_k \binom{n}{k} F_{m+k} &= F_{m+2n} \end{aligned}$$

but these identities have already been proved using the addition operator S.

6.11 Definite and Indefinite summation

The following result is one of the most important rule connecting the finite operator method and combinatorial sums:

$$\begin{aligned} \sum_{k=0}^n E^k &= [z^n]\frac{1}{1-z}\frac{1}{1-Ez} = \\ &= [z^n]\frac{1}{(E-1)z}\left(\frac{1}{1-Ez}-\frac{1}{1-z}\right) = \\ &= \frac{1}{E-1}[z^{n+1}]\left(\frac{1}{1-Ez}-\frac{1}{1-z}\right) = \\ &= \frac{E^{n+1}-1}{E-1} = (E^{n+1}-1)\Delta^{-1} \end{aligned}$$

We observe that the operator E commutes with the indeterminate z, which is constant with respect to the variable x, on which E operates. The rule above is called the rule of *definite summation*; the operator Δ^{-1} is called *indefinite summation* and is often denoted by Σ. In order to make this point clear, let us consider any function $f(x)$ and suppose that a function $g(x)$ exists such that $\Delta g(x) = f(x)$. Hence we have $\Delta^{-1} f(x) = g(x)$ and the rule of definite summation immediately gives:

$$\sum_{k=0}^n f(x+k) = g(x+n+1) - g(x)$$

This is analogous to the rule of definite integration. In fact, the operator of indefinite integration $\int dx$ is inverse of the differentiation operator D, and if $f(x)$ is any function, a primitive function for $f(x)$ is any function $\hat{g}(x)$ such that $D\hat{g}(x) = f(x)$ or $D^{-1}f(x) = \int f(x)dx = \hat{g}(x)$. The fundamental theorem of the integral calculus relates definite and indefinite integration:

$$\int_a^b f(x)dx = \hat{g}(b) - \hat{g}(a)$$

The formula for definite summation can be written in a similar way, if we consider the integer variable k and set $a = x$ and $b = x+n+1$:

$$\sum_{k=a}^{b-1} f(k) = g(b) - g(a)$$

These facts create an analogy between Δ^{-1} and D^{-1}, or Σ and $\int dx$, which can be stressed by considering the formal properties of Σ. First of all, we observe that $g(x) = \Sigma f(x)$ is not uniquely determined. If $C(x)$ is any function periodic of period 1, i.e., $C(x+k) = C(x), \forall k \in \mathbb{Z}$, we have:

$$\Delta(g(x)+C(x)) = \Delta g(x) + \Delta C(x) = \Delta g(x)$$

and therefore:

$$\Sigma f(x) = g(x) + C(x)$$

When $f(x)$ is a sequence and only is defined for integer values of x, the function $C(x)$ reduces to a constant, and plays the same rôle as the integration constant in the operation of indefinite integration.

The operator Σ is obviously linear:

$$\Sigma(\alpha f(x) + \beta g(x)) = \alpha\Sigma f(x) + \beta\Sigma g(x)$$

This is proved by applying the operator Δ to both sides.

An important property is *summation by parts*, corresponding to the well-known rule of the indefinite integration operator. Let us begin with the rule for the difference of a product, which we proved earlier:

$$\Delta(f(x)g(x)) = \Delta f(x) E g(x) + f(x)\Delta g(x)$$

By applying the operator Σ to both sides and exchanging terms:

$$\Sigma(f(x)\Delta g(x)) = f(x)g(x) - \Sigma(\Delta f(x) E g(x))$$

This formula allows us to change a sum of products, of which we know that the second factor is a difference, into a sum involving the difference of the first factor. The transformation can be convenient every time when the difference of the first factor is simpler than the difference of the second factor. For example, let us perform the following indefinite summation:

$$\begin{aligned}\Sigma x\binom{x}{m} &= \Sigma x\Delta\binom{x}{m+1} = \\ &= x\binom{x}{m+1} - \Sigma(\Delta x)\binom{x+1}{m+1} = \\ &= x\binom{x}{m+1} - \Sigma\binom{x+1}{m+1} = \\ &= x\binom{x}{m+1} - \binom{x+1}{m+2}\end{aligned}$$

Obviously, this indefinite sum can be transformed into a definite sum by using the first result in this section:

$$\begin{aligned}\sum_{k=a}^{b} k\binom{k}{m} &= (b+1)\binom{b+1}{m+1} - \\ &- a\binom{a}{m+1} - \binom{b+2}{m+2} + \binom{a+1}{m+2}\end{aligned}$$

and for $a = 0$ and $b = n$:

$$\sum_{k=0}^{n} k\binom{k}{m} = (n+1)\binom{n+1}{m+1} - \binom{n+2}{m+2}$$

6.12 Definite Summation

In a sense, the rule:

$$\sum_{k=0}^{n} E^k = (E^{n+1} - 1)\Delta^{-1} = (E^{n+1} - 1)\Sigma$$

is the most important result of the operator method. In fact, it reduces the sum of the successive elements in a sequence to the computation of the indefinite sum, and this is just the operator inverse of the difference. Unfortunately, Δ^{-1} is not easy to compute and, apart from a restricted number of cases, there is no general rule allowing us to guess what $\Delta^{-1}f(x) = \Sigma f(x)$ might be. In this rather pessimistic sense, the rule is very fine, very general and completely useless.

However, from a more positive point of view, we can say that whenever we know, in some way or another, an expression for $\Delta^{-1}f(x) = \Sigma f(x)$, we have solved the problem of finding $\sum_{k=0}^{n} f(x+k)$. For example, we can look at the differences computed in the previous sections and, for each of them, obtain the Σ of some function; in this way we immediately have a number of sums. The negative point is that, sometimes, we do not have a simple function and, therefore, the sum may not have any combinatorial interest.

Here is a number of identities obtained by our previous computations.

1) We have again the partial sums of the geometric series:

$$\begin{aligned}\Delta^{-1}p^x &= \frac{p^x}{p-1} = \Sigma p^x \\ \sum_{k=0}^{n} p^{x+k} &= \frac{p^{x+n+1} - p^x}{p-1} \\ \sum_{k=0}^{n} p^k &= \frac{p^{n+1}-1}{p-1} \qquad (x=0)\end{aligned}$$

2) The sum of consecutive Fibonacci numbers:

$$\begin{aligned}\Delta^{-1}F_x &= F_{x+1} = \Sigma F_x \\ \sum_{k=0}^{n} F_{x+k} &= F_{x+n+2} - F_{x+1} \\ \sum_{k=0}^{n} F_k &= F_{n+2} - 1 \qquad (x=0)\end{aligned}$$

3) The sum of consecutive binomial coefficients with constant denominator:

$$\begin{aligned}\Delta^{-1}\binom{x}{m} &= \binom{x}{m+1} = \Sigma\binom{x}{m} \\ \sum_{k=0}^{n}\binom{x+k}{m} &= \binom{x+n+1}{m+1} - \binom{x}{m+1} \\ \sum_{k=0}^{n}\binom{k}{m} &= \binom{n+1}{m+1} \qquad (x=0)\end{aligned}$$

4) The sum of consecutive binomial coefficients:

$$\begin{aligned}\Delta^{-1}\binom{p+x}{m+x} &= \binom{p+x}{m+x-1} = \Sigma\binom{p+x}{m+x} \\ \sum_{k=0}^{n}\binom{p+k}{m+k} &= \binom{p+n+1}{m+n} - \binom{p}{m-1}\end{aligned}$$

5) The sum of falling factorials:

$$\begin{aligned}\Delta^{-1}x^{\underline{m}} &= \frac{1}{m+1}x^{\underline{m+1}} = \Sigma x^{\underline{m}} \\ \sum_{k=0}^{n}(x+k)^{\underline{m}} &= \frac{(x+n+1)^{\underline{m+1}} - x^{\underline{m+1}}}{m+1} \\ \sum_{k=0}^{n} k^{\underline{m}} &= \frac{1}{m+1}(n+1)^{\underline{m+1}} \qquad (x=0).\end{aligned}$$

6) The sum of raising factorials:

$$\Delta^{-1}x^{\overline{m}} = \frac{1}{m+1}(x-1)^{\overline{m+1}} = \Sigma x^{\overline{m}}$$

$$\sum_{k=0}^{n}(x+k)^{\overline{m}} = \frac{(x+n)^{\overline{m+1}}-(x-1)^{\overline{m+1}}}{m+1}$$

$$\sum_{k=0}^{n}k^{\overline{m}} = \frac{1}{m+1}n^{\overline{m+1}} \qquad (x=0).$$

7) The sum of inverse binomial coefficients:

$$\Delta^{-1}\binom{x}{m}^{-1} = \frac{m}{m-1}\binom{x-1}{m-1}^{-1} = \Sigma\binom{x}{m}$$

$$\sum_{k=0}^{n}\binom{x+k}{m}^{-1} = \frac{m}{m-1}\left(\binom{x-1}{m-1}^{-1}-\binom{x+n}{m-1}^{-1}\right).$$

8) The sum of harmonic numbers. Since $1=\Delta x$, we have:

$$\begin{aligned}\Delta^{-1}H_x &= xH_x - x = \Sigma H_x\\ \sum_{k=0}^{n}H_{x+k} &= (x+n+1)H_{x+n+1} - xH_x - (n+1)\\ \sum_{k=0}^{n}H_k &= (n+1)H_{n+1}-(n+1) =\\ &= (n+1)H_n - n \qquad (x=0).\end{aligned}$$

6.13 The Euler-McLaurin Summation Formula

One of the most striking applications of the finite operator method is the formal proof of the *Euler-McLaurin summation formula*. The starting point is the Taylor theorem for the series expansion of a function $f(x)\in C^\infty$, i.e., a function having derivatives of any order. The usual form of the theorem:

$$\begin{aligned}f(x+h) &= f(x)+\frac{h}{1!}f'(x)+\frac{h^2}{2!}f''(x)+\\ &\quad +\cdots+\frac{h^n}{n!}f^{(n)}(x)+\cdots\end{aligned}$$

can be interpreted in the sense of operators as a result connecting the shift and the differentiation operators. In fact, for $h=1$, it can be written as:

$$Ef(x) = If(x)+\frac{Df(x)}{1!}+\frac{D^2f(x)}{2!}+\cdots$$

and therefore as a relation between operators:

$$E = 1+\frac{D}{1!}+\frac{D^2}{2!}+\cdots+\frac{D^n}{n!}+\cdots = e^D$$

This formal identity relates the finite operator E and the infinitesimal operator D, and subtracting 1 from both sides it can be formulated as:

$$\Delta = e^D - 1$$

By inverting, we have a formula for the Σ operator:

$$\Sigma = \frac{1}{e^D-1} = \frac{1}{D}\left(\frac{D}{e^D-1}\right)$$

Now, we recognize the generating function of the Bernoulli numbers, and therefore we have a development for Σ:

$$\begin{aligned}\Sigma &= \frac{1}{D}\left(B_0+\frac{B_1}{1!}D+\frac{B_2}{2!}D^2+\cdots\right) =\\ &= D^{-1}-\frac{1}{2}I+\frac{1}{12}D-\frac{1}{720}D^3+\\ &\quad +\frac{1}{30240}D^5-\frac{1}{1209600}D^7+\cdots.\end{aligned}$$

This is not a series development since, as we know, the Bernoulli numbers diverge to infinity. We have a case of *asymptotic development*, which only is defined when we consider a limited number of terms, but in general diverges if we let the number of terms go to infinity. The number of terms for which the sum approaches its true value depends on the function $f(x)$ and on the argument x.

From the indefinite we can pass to the definite sum by applying the general rule of Section 6.12. Since $D^{-1}=\int dx$, we immediately have:

$$\begin{aligned}\sum_{k=0}^{n-1}f(k) &= \int_0^n f(x)\,dx-\frac{1}{2}\left[f(x)\right]_0^n+\\ &\quad +\frac{1}{12}\left[f'(x)\right]_0^n-\frac{1}{720}\left[f'''(x)\right]_0^n+\cdots\end{aligned}$$

and this is the celebrated Euler-McLaurin summation formula. It expresses a sum as a function of the integral and the successive derivatives of the function $f(x)$. In this sense, the formula can be seen as a method for approximating a sum by means of an integral or, vice versa, for approximating an integral by means of a sum, and this was just the point of view of the mathematicians who first developed it.

As a simple but very important example, let us find an asymptotic development for the harmonic numbers H_n. Since $H_n = H_{n-1}+1/n$, the Euler-McLaurin formula applies to H_{n-1} and to the function $f(x)=1/x$, giving:

$$\begin{aligned}H_{n-1} &= \int_1^n\frac{dx}{x}-\frac{1}{2}\left[\frac{1}{x}\right]_1^n+\frac{1}{12}\left[-\frac{1}{x^2}\right]_1^n-\\ &\quad -\frac{1}{720}\left[-\frac{6}{x^4}\right]_1^n+\frac{1}{30240}\left[-\frac{120}{x^6}\right]_1^n+\cdots\\ &= \ln n-\frac{1}{2n}+\frac{1}{2}-\frac{1}{12n^2}+\frac{1}{12}+\frac{1}{120n^4}-\\ &\quad -\frac{1}{120}-\frac{1}{256n^6}+\frac{1}{252}+\cdots\end{aligned}$$

In this expression a number of constants appears, and they can be summed together to form a constant γ, provided that the sum actually converges. However, we observe that as $n \to \infty$ this constant γ is the Euler-Mascheroni constant:

$$\lim_{n\to\infty} (H_{n-1} - \ln n) = \gamma = 0.577215664902\ldots$$

By adding $1/n$ to both sides of the previous relation, we eventually find:

$$H_n = \ln n + \gamma + \frac{1}{2n} - \frac{1}{12n^2} + \frac{1}{120n^4} - \frac{1}{252n^6} + \cdots$$

and this is the asymptotic expansion we were looking for.

6.14 Applications of the Euler-McLaurin Formula

As another application of the Euler-McLaurin summation formula, we now show the derivation of the Stirling's approximation for $n!$. The first step consists in taking the logarithm of that quantity:

$$\ln n! = \ln 1 + \ln 2 + \ln 3 + \cdots + \ln n$$

so that we are reduced to compute a sum and hence to apply the Euler-McLaurin formula:

$$\begin{aligned}
\ln(n-1)! &= \sum_{k=1}^{n-1} \ln k = \int_1^n \ln x\, dx - \frac{1}{2}\left[\ln x\right]_1^n + \\
&\quad + \frac{1}{12}\left[\frac{1}{x}\right]_1^n - \frac{1}{720}\left[\frac{2}{x^3}\right]_1^n + \cdots = \\
&= n\ln n - n + 1 - \frac{1}{2}\ln n + \frac{1}{12n} - \\
&\quad - \frac{1}{12} - \frac{1}{360n^2} + \frac{1}{360} + \cdots .
\end{aligned}$$

Here we have used the fact that $\int \ln x\, dx = x\ln x - x$. At this point we can add $\ln n$ to both sides and introduce a constant $\sigma = 1 - 1/12 + 1/360 - \cdots$ It is not by all means easy to determine directly the value of σ, but by other approaches to the same problem it is known that $\sigma = \ln\sqrt{2\pi}$. Numerically, we can observe that:

$$1 - \frac{1}{12} + \frac{1}{360} = 0.919(4) \quad \text{and} \quad \ln\sqrt{2\pi} \approx 0.9189388.$$

We can now go on with our sum:

$$\ln n! = n\ln n - n + \frac{1}{2}\ln n + \ln\sqrt{2\pi} + \frac{1}{12n} - \frac{1}{360n^3} + \cdots$$

To obtain the value of $n!$ we only have to take exponentials:

$$\begin{aligned}
n! &= \frac{n^n}{e^n}\sqrt{n}\sqrt{2\pi}\exp\left(\frac{1}{12n}\right)\exp\left(-\frac{1}{360n^3}\right)\cdots \\
&= \sqrt{2\pi n}\frac{n^n}{e^n}\left(1 + \frac{1}{12n} + \frac{1}{288n^2} + \cdots\right)\times \\
&\quad \times\left(1 - \frac{1}{360n^3} + \cdots\right)\cdots = \\
&= \sqrt{2\pi n}\frac{n^n}{e^n}\left(1 + \frac{1}{12n} + \frac{1}{288n^2} - \cdots\right).
\end{aligned}$$

This is the well-known Stirling's approximation for $n!$. By means of this approximation, we can also find the approximation for another important quantity:

$$\begin{aligned}
\binom{2n}{n} &= \frac{(2n)!}{n!^2} = \frac{\sqrt{4\pi n}\left(\frac{2n}{e}\right)^{2n}}{2\pi n\left(\frac{n}{e}\right)^{2n}} \times \\
&\quad \times \frac{1 + \frac{1}{24n} + \frac{1}{1142n^2} - \cdots}{\left(1 + \frac{1}{12n} + \frac{1}{288n^2} - \cdots\right)^2} = \\
&= \frac{4^n}{\sqrt{\pi n}}\left(1 - \frac{1}{8n} + \frac{1}{128n^2} + \cdots\right).
\end{aligned}$$

Another application of the Euler-McLaurin summation formula is given by the sum $\sum_{k=1}^n k^p$, when p is any integer constant different from -1, which is the case of the harmonic numbers:

$$\begin{aligned}
\sum_{k=0}^{n-1} k^p &= \int_0^n x^p\, dx - \frac{1}{2}\left[x^p\right]_0^n + \frac{1}{12}\left[px^{p-1}\right]_0^n - \\
&\quad - \frac{1}{720}\left[p(p-1)(p-2)x^{p-3}\right]_0^n + \cdots = \\
&= \frac{n^{p+1}}{p+1} - \frac{n^p}{2} + \frac{pn^{p-1}}{12} - \\
&\quad - \frac{p(p-1)(p-2)n^{p-3}}{720} + \cdots .
\end{aligned}$$

In this case the evaluation at 0 does not introduce any constant. By adding n^p to both sides, we have the following formula, which only contains a finite number of terms:

$$\begin{aligned}
\sum_{k=0}^{n} k^p &= \frac{n^{p+1}}{p+1} + \frac{n^p}{2} + \frac{pn^{p-1}}{12} - \\
&\quad - \frac{p(p-1)(p-2)n^{p-3}}{720} + \cdots
\end{aligned}$$

If p is not an integer, after $\lceil p \rceil$ differentiations, we obtain x^q, where $q < 0$, and therefore we cannot consider the limit 0. We proceed with the Euler-McLaurin formula in the following way:

$$\begin{aligned}
\sum_{k=1}^{n-1} k^p &= \int_1^n x^p\, dx - \frac{1}{2}\left[x^p\right]_1^n + \frac{1}{12}\left[px^{p-1}\right]_1^n - \\
&\quad - \frac{1}{720}\left[p(p-1)(p-2)x^{p-3}\right]_1^n + \cdots = \\
&= \frac{n^{p+1}}{p+1} - \frac{1}{p+1} - \frac{1}{2}n^p + \frac{1}{2} + \frac{pn^{p-1}}{12} - \\
&\quad - \frac{p}{12} - \frac{p(p-1)(p-2)n^{p-3}}{720} +
\end{aligned}$$

$$
\begin{aligned}
& + \frac{p(p-1)(p-2)}{720}\cdots \\
\sum_{k=1}^{n} k^p \quad = \quad & \frac{n^{p+1}}{p+1} + \frac{n^p}{2} + \frac{pn^{p-1}}{12} - \\
& - \frac{p(p-1)(p-2)n^{p-3}}{720} + \cdots + K_p.
\end{aligned}
$$

The constant:

$$K_p = -\frac{1}{p+1} + \frac{1}{2} - \frac{p}{12} + \frac{p(p-1)(p-2)}{720} + \cdots$$

has a fundamental rôle when the leading term $n^{p+1}/(p+1)$ does not increase with n, i.e., when $p < -1$. In that case, in fact, the sum converges to K_p. When $p > -1$ the constant is less important. For example, we have:

$$\sum_{k=1}^{n} \sqrt{k} = \frac{2}{3}n\sqrt{n} + \frac{\sqrt{n}}{2} + K_{1/2} + \frac{1}{24\sqrt{n}} + \cdots$$

$$K_{1/2} \approx -0.2078862\ldots$$

$$\sum_{k=1}^{n} \frac{1}{\sqrt{k}} = 2\sqrt{n} + K_{-1/2} + \frac{1}{2\sqrt{n}} - \frac{1}{24n\sqrt{n}} + \cdots$$

$$K_{-1/2} \approx -1.4603545\ldots$$

For $p = -2$ we find:

$$\sum_{k=1}^{n} \frac{1}{k^2} = K_{-2} - \frac{1}{n} + \frac{1}{2n^2} - \frac{1}{6n^3} + \frac{1}{30n^5} - \cdots$$

It is possible to show that $K_{-2} = \pi^2/6$ and therefore we have a way to approximate the sum (see Section 2.7).

Chapter 7

Asymptotics

7.1 The convergence of power series

In many occasions, we have pointed out that our approach to power series was purely formal. Because of that, we always spoke of "formal power series", and never considered convergence problems. As we have seen, a lot of things can be said about formal power series, but now the moment has arrived that we must turn to talk about the convergence of power series. We will see that this allows us to evaluate the asymptotic behavior of the coefficients f_n of a power series $\sum_n f_n t^n$, thus solving many problems in which the exact value of f_n cannot be found. In fact, many times, the asymptotic evaluation of f_n can be made more precise and an actual approximation of f_n can be found.

The natural setting for talking about convergence is the field $\mathbb{C}$ of the complex numbers and therefore, from now on, we will think of the indeterminate t as of a variable taking its values from $\mathbb{C}$. Obviously, a power series $f(t) = \sum_n f_n t^n$ *converges* for some $t_0 \in \mathbb{C}$ iff $f(t_0) = \sum_n f_n t_0^n < \infty$, and *diverges* iff $\lim_{t\to t_0} f(t) = \infty$. There are cases for which a series neither converges nor diverges; for example, when $t = 1$, the series $\sum_n (-1)^n t^n$ does not tend to any limit, finite or infinite. Therefore, when we say that a series *does not converge* (to a finite value) for a given value $t_0 \in \mathbb{C}$, we mean that the series in t_0 diverges or does not tend to any limit.

A basic result on convergence is given by the following:

Theorem 7.1.1 *Let $f(t) = \sum_n f_n t^n$ be a power series such that $f(t_0)$ converges for the value $t_0 \in \mathbb{C}$. Then $f(t)$ converges for every $t_1 \in \mathbb{C}$ such that $|t_1| < |t_0|$.*

Proof: If $f(t_0) < \infty$ then an index $N \in \mathbb{N}$ exists such that for every $n > N$ we have $|f_n t_0^n| \leq |f_n||t_0|^n < M$, for some finite $M \in \mathbb{R}$. This means $|f_n| < M/|t_0|^n$ and therefore:

$$\left|\sum_{n=N}^{\infty} f_n t_1^n\right| \leq \sum_{n=N}^{\infty} |f_n||t_1|^n \leq$$

$$\leq \sum_{n=N}^{\infty} \frac{M}{|t_0|^n}|t_1|^n = M \sum_{n=N}^{\infty} \left(\frac{|t_1|}{|t_0|}\right)^n < \infty$$

because the last sum is a geometric series with $|t_1|/|t_0| < 1$ by the hypothesis $|t_1| < |t_0|$. Since the first N terms obviously amount to a finite quantity, the theorem follows. ∎

In a similar way, we can prove that if the series diverges for some value $t_0 \in \mathbb{C}$, then it diverges for every value t_1 such that $|t_1| > |t_0|$. Obviously, a series can converge for the single value $t_0 = 0$, as it happens for $\sum_n n! t^n$, or can converge for every value $t \in \mathbb{C}$, as for $\sum_n t^n/n! = e^t$. In all the other cases, the previous theorem implies:

Theorem 7.1.2 *Let $f(t) = \sum_n f_n t^n$ be a power series; then there exists a non-negative number $R \in \mathbb{R}$ or $R = \infty$ such that:*

1. *for every complex number t_0 such that $|t_0| < R$ the series (absolutely) converges and, in fact, the convergence is uniform in every circle of radius $\rho < R$;*
2. *for every complex number t_0 such that $|t_0| > R$ the series does not converge.*

The uniform convergence derives from the previous proof: the constant M can be made unique by choosing the largest value for all the t_0 such that $|t_0| \leq \rho$.

The value of R is uniquely determined and is called the *radius of convergence* for the series. From the proof of the theorem, for $r < R$ we have $|f_n| r^n \leq M$ or $\sqrt[n]{|f_n|} \leq \sqrt[n]{M}/r \to 1/r$; this implies that $\limsup \sqrt[n]{|f_n|} \leq 1/R$. Besides, for $r > R$ we have $|f_n| r^n \geq 1$ for infinitely many n; this implies $\limsup \sqrt[n]{|f_n|} \geq 1/R$, and therefore we have the following formula for the radius of convergence:

$$\frac{1}{R} = \limsup_{n\to\infty} \sqrt[n]{|f_n|}.$$

This result is the basis for our considerations on the asymptotics of a power series coefficients. In fact, it implies that, as a first approximation, $|f_n|$ grows as $1/R^n$. However, this is a rough estimate, because it can also grow as n/R^n or $1/(nR^n)$, and many possibilities arise, which can make more precise the basic approximation; the next sections will be dedicated to this problem. We conclude by noticing that if:

$$\lim_{n\to\infty}\left|\frac{f_{n+1}}{f_n}\right| = S$$

then $R = 1/S$ is the radius of convergence of the series.

7.2 The method of Darboux

Newton's rule is the basis for many considerations on asymptotics. In practice, we used it to prove that $F_n \sim \phi^n/\sqrt{5}$, and many other proofs can be performed by using Newton's rule together with the following theorem, whose relevance was noted by Bender and, therefore, will be called *Bender's theorem*:

Theorem 7.2.1 *Let $f(t) = g(t)h(t)$ where $f(t)$, $g(t)$, $h(t)$ are power series and $h(t)$ has a radius of convergence larger than $f(t)$'s (which therefore equals the radius of convergence of $g(t)$); if $\lim_{n\to\infty} g_n/g_{n+1} = b$ and $h(b) \neq 0$, then:*

$$f_n \sim h(b)g_n$$

Let us remember that if $g(t)$ has positive real coefficients, then g_n/g_{n+1} tends to the radius of convergence of $g(t)$. The proof of this theorem is omitted here; instead, we give a simple example. Let us suppose we wish to find the asymptotic value for the Motzkin numbers, whose generating function is:

$$\mu(t) = \frac{1-t-\sqrt{1-2t-3t^2}}{2t^2}.$$

For $n \geq 2$ we obviously have:

$$\mu_n = [t^n]\frac{1-t-\sqrt{1-2t-3t^2}}{2t^2} =$$
$$= -\frac{1}{2}[t^{n+2}]\sqrt{1+t}\,(1-3t)^{1/2}.$$

We now observe that the radius of convergence of $\mu(t)$ is $R = 1/3$, which is the same as the radius of $g(t) = (1-3t)^{1/2}$, while $h(t) = \sqrt{1+t}$ has 1 as radius of convergence; therefore we have $\mu_n/\mu_{n+1} \to 1/3$ as $n \to \infty$. By Bender's theorem we find:

$$\begin{aligned}\mu_n &\sim -\frac{1}{2}\sqrt{\frac{4}{3}}\,[t^{n+2}](1-3t)^{1/2} = \\ &= -\frac{\sqrt{3}}{3}\binom{1/2}{n+2}(-3)^{n+2} = \\ &= \frac{\sqrt{3}}{3(2n+3)}\binom{2n+4}{n+2}\left(\frac{3}{4}\right)^{n+2}.\end{aligned}$$

This is a particular case of a more general result due to Darboux and known as *Darboux' method*. First of all, let us show how it is possible to obtain an approximation for the binomial coefficient $\binom{\gamma}{n}$, when $\gamma \in \mathbb{C}$ is a fixed number and n is large. We begin by proving the following formula for the ratio of two large values of the Γ function (a, b are two small parameters with respect to n):

$$\begin{aligned}&\frac{\Gamma(n+a)}{\Gamma(n+b)} = \\ &\quad = n^{a-b}\left(1 + \frac{(a-b)(a+b-1)}{2n} + O\left(\frac{1}{n^2}\right)\right).\end{aligned}$$

Let us apply the Stirling formula for the Γ function:

$$\begin{aligned}&\frac{\Gamma(n+a)}{\Gamma(n+b)} \approx \\ &\quad \approx \sqrt{\frac{2\pi}{n+a}}\left(\frac{n+a}{e}\right)^{n+a}\left(1+\frac{1}{12(n+a)}\right)\times \\ &\qquad \times\sqrt{\frac{n+b}{2\pi}}\left(\frac{e}{n+b}\right)^{n+b}\left(1-\frac{1}{12(n+b)}\right).\end{aligned}$$

If we limit ourselves to the term in $1/n$, the two corrections cancellate each other and therefore we find:

$$\begin{aligned}\frac{\Gamma(n+a)}{\Gamma(n+b)} &\approx \sqrt{\frac{n+b}{n+a}}e^{b-a}\frac{(n+a)^{n+a}}{(n+b)^{n+b}} = \\ &= \sqrt{\frac{n+b}{n+a}}e^{b-a}n^{a-b}\frac{(1+a/n)^{n+a}}{(1+b/n)^{n+b}}.\end{aligned}$$

We now obtain asymptotic approximations in the following way:

$$\begin{aligned}\sqrt{\frac{n+b}{n+a}} &= \frac{\sqrt{1+b/n}}{\sqrt{1+a/n}} \approx \\ &\approx \left(1+\frac{b}{2n}\right)\left(1-\frac{a}{2n}\right) \approx 1+\frac{b-a}{2n}.\end{aligned}$$

$$\begin{aligned}\left(1+\frac{x}{n}\right)^{n+x} &= \exp\left((n+x)\ln\left(1+\frac{x}{n}\right)\right) = \\ &= \exp\left((n+x)\left(\frac{x}{n}-\frac{x^2}{2n^2}+\cdots\right)\right) = \\ &= \exp\left(x+\frac{x^2}{n}-\frac{x^2}{2n}+\cdots\right) = \\ &= e^x\left(1+\frac{x^2}{2n}+\cdots\right).\end{aligned}$$

Therefore, for our expression we have:

$$\begin{aligned}&\frac{\Gamma(n+a)}{\Gamma(n+b)} \approx \\ &\quad \approx \frac{n^{a-b}}{e^{a-b}}\frac{e^a}{e^b}\left(1+\frac{a^2}{2n}\right)\left(1-\frac{b^2}{2n}\right)\left(1+\frac{b-a}{2n}\right) = \\ &\quad = n^{a-b}\left(1+\frac{a^2-b^2-a+b}{2n}+O\left(\frac{1}{n^2}\right)\right).\end{aligned}$$

We are now in a position to prove the following:

Theorem 7.2.2 *Let $f(t) = h(t)(1-\alpha t)^{\gamma}$, for some γ which is not a positive integer, and $h(t)$ having a radius of convergence larger than $1/\alpha$. Then we have:*

$$f_n = [t^n]f(t) \sim h\left(\frac{1}{\alpha}\right)\binom{\gamma}{n}(-\alpha)^n = \frac{\alpha^n h(1/\alpha)}{\Gamma(-\gamma)n^{1+\gamma}}.$$

Proof: We simply apply Bender's theorem and the formula for approximating the binomial coefficient:

$$\binom{\gamma}{n} = \frac{\gamma(\gamma-1)\cdots(\gamma-n+1)}{n!} =$$

$$= \frac{(-1)^n(n-\gamma-1)(n-\gamma-2)\cdots(1-\gamma)(-\gamma)}{\Gamma(n+1)}.$$

By repeated applications of the recurrence formula for the Γ function $\Gamma(x+1) = x\Gamma(x)$, we find:

$$\Gamma(n-\gamma) =$$

$$= (n-\gamma-1)(n-\gamma-2)\cdots(1-\gamma)(-\gamma)\Gamma(-\gamma)$$

and therefore:

$$\begin{aligned}\binom{\gamma}{n} &= \frac{(-1)^n\Gamma(n-\gamma)}{\Gamma(n+1)\Gamma(-\gamma)} = \\ &= \frac{(-1)^n}{\Gamma(-\gamma)}n^{-1-\gamma}\left(1+\frac{\gamma(\gamma+1)}{2n}\right)\end{aligned}$$

from which the desired formula follows. ∎

7.3 Singularities: poles

The considerations in the previous sections show how important is to determine the radius of convergence of a series, when we wish to have an approximation of its coefficients. Therefore, we are now going to look more closely to methods for finding the radius of convergence of a given function $f(t)$. In order to do this, we have to distinguish between the function $f(t)$ and its series development, which will be denoted by $\widehat{f}(t)$. So, for example, we have $f(t) = 1/(1-t)$ and $\widehat{f}(t) = 1+t+t^2+t^3+\cdots$. The series $\widehat{f}(t)$ represents $f(t)$ inside the circle of convergence, in the sense that $\widehat{f}(t) = f(t)$ for every t internal to this circle, but $\widehat{f}(t)$ can be different from $f(t)$ on the border of the circle or outside it, where actually the series does not converge. Therefore, the radius of convergence can be determined if we are able to find out values t_0 for which $f(t_0) \neq \widehat{f}(t_0)$.

A first case is when t_0 is an isolated point for which $\lim_{t\to t_0} f(t) = \infty$. In fact, in this case, for every t such that $|t| > |t_0|$ the series $\widehat{f}(t)$ should diverge as we have seen in the previous section, and therefore $\widehat{f}(t)$ must be different from $f(t)$. This is the case of $f(t) = 1/(1-t)$, which goes to ∞ when $t \to 1$. When $|t| > 1$, $f(t)$ assumes a well-defined value while $\widehat{f}(t)$ diverges.

We will call a *singularity* for $f(t)$ every point $t_0 \in \mathbb{C}$ such that in every neighborhood of t_0 there is a t for which $f(t)$ and $\widehat{f}(t)$ behaves differently and a t' for which $f(t') = \widehat{f}(t')$. Therefore, $t_0 = 1$ is a singularity for $f(t) = 1/(1-t)$. Because our previous considerations, the singularities of $f(t)$ determine its radius of convergence; on the other hand, no singularity can be contained in the circle of convergence, and therefore the radius of convergence is determined by the singularity or singularities of smallest modulus. These will be called *dominating singularities* and we observe explicitly that a function can have more than one dominating singularity. For example, $f(t) = 1/(1-t^2)$ has $t = 1$ and $t = -1$ as dominating singularities, because $|1| = |-1|$. The radius of convergence is always a non-negative real number and we have $R = |t_0|$, if t_0 is any one of the dominating singularities for $f(t)$.

An isolated point t_0 for which $f(t_0) = \infty$ is therefore a singularity for $f(t)$; as we shall see, not every singularity of $f(t)$ is such that $f(t) = \infty$, but, for the moment, let us limit ourselves to this case. The following situation is very important: if $f(t_0) = \infty$ and we set $\alpha = 1/t_0$, we will say that t_0 is a *pole* for $f(t)$ iff there exists a positive integer m such that:

$$\lim_{t\to t_0}(1-\alpha t)^m f(t) = K < \infty \quad \text{and} \quad K \neq 0.$$

The integer m is called the *order* of the pole. By this definition, the function $f(t) = 1/(1-t)$ has a pole of order 1 in $t_0 = 1$, while $1/(1-t)^2$ has a pole of order 2 in $t_0 = 1$ and $1/(1-2t)^5$ has a pole of order 5 in $t_0 = 1/2$. A more interesting case is $f(t) = (e^t - e)/(1-t)^2$, which, notwithstanding the $(1-t)^2$, has a pole of order 1 in $t_0 = 1$; in fact:

$$\lim_{t\to 1}(1-t)\frac{e^t-e}{(1-t)^2} = \lim_{t\to 1}\frac{e^t-e}{1-t} = \lim_{t\to 1}\frac{e^t}{-1} = -e.$$

The generating function of Bernoulli numbers $f(t) = t/(e^t-1)$ has infinitely many poles. Observe first that $t = 0$ is not a pole because:

$$\lim_{t\to 0}\frac{t}{e^t-1} = \lim_{t\to 0}\frac{1}{e^t} = 1.$$

The denominator becomes 0 when $e^t = 1$, and this happens when $t = 2k\pi i$; in fact, $e^{2k\pi i} = \cos 2k\pi + i\sin 2k\pi = 1$. In that case, the dominating singularities are $t_0 = 2\pi i$ and $t_1 = -2\pi i$. Finally, the generating function of the ordered Bell numbers $f(t) = 1/(2-e^t)$ has again an infinite number of poles $t = \ln 2 + 2k\pi i$; in this case the dominating singularity is $t_0 = \ln 2$.

We conclude this section by observing that if $f(t_0) = \infty$, not necessarily t_0 is a pole for $f(t)$. In

fact, let us consider the generating function for the central binomial coefficients $f(t) = 1/\sqrt{1-4t}$. For $t_0 = 1/4$ we have $f(1/4) = \infty$, but t_0 is not a pole of order 1 because:

$$\lim_{t\to 1/4} \frac{1-4t}{\sqrt{1-4t}} = \lim_{t\to 1/4} \sqrt{1-4t} = 0$$

and the same happens if we try with $(1-4t)^m$ for $m > 1$. As we shall see, this kind of singularity is called "algebraic". Finally, let us consider the function $f(t) = \exp(1/(1-t))$, which goes to ∞ as $t \to 1$. In this case we have:

$$\lim_{t\to 1}(1-t)^m \exp\left(\frac{1}{1-t}\right) =$$

$$= \lim_{t\to 1}(1-t)^m \left(1 + \frac{1}{1-t} + \frac{1}{2(1-t)^2} + \cdots\right) = \infty.$$

In fact, the first $m-1$ terms tend to 0, the mth term tends to $1/m!$, but all the other terms go to ∞. Therefore, $t_0 = 1$ is not a pole of any order. Whenever we have a function $f(t)$ for which a point $t_0 \in \mathbb{C}$ exists such that $\forall m > 0$: $\lim_{t\to t_0}(1-t/t_0)^m f(t) = \infty$, we say that t_0 is an *essential singularity* for $f(t)$. Essential singularities are points at which $f(t)$ goes to ∞ too fast; these singularities cannot be treated by Darboux' method and their study will be delayed until we study the Hayman's method.

7.4 Poles and asymptotics

Darboux' method can be easily used to deal with functions, whose dominating singularities are poles. Actually, a direct application of Bender's theorem is sufficient, and this is the way we will use in the following examples.

Fibonacci numbers are easily approximated:

$$[t^n]\frac{t}{1-t-t^2} = [t^n]\frac{t}{1-\widehat{\phi}t}\frac{1}{1-\phi t} \sim$$

$$\sim \quad \left[\frac{t}{1-\widehat{\phi}t} \;\middle|\; t = \frac{1}{\phi}\right][t^n]\frac{1}{1-\phi t} = \frac{1}{\sqrt{5}}\phi^n.$$

Our second example concerns a particular kind of permutations, called *derangements* (see Section 2.2). A derangement is a permutation without any fixed point. For $n = 0$ the empty permutation is considered a derangement, since no fixed point exists. For $n = 1$, there is no derangement, but for $n = 2$ the permutation $(1\,2)$, written in cycle notation, is actually a derangement. For $n = 3$ we have the two derangements $(1\,2\,3)$ and $(1\,3\,2)$, and for $n = 4$ we have a total of 9 derangements.

Let D_n the number of derangements in P_n; we can count them in the following way: we begin by subtracting from $n!$, the total number of permutations, the number of permutations having at least a fixed point: if the fixed point is 1, we have $(n-1)!$ possible permutations; if the fixed point is 2, we have again $(n-1)!$ permutations of the other elements. Therefore, we have a total of $n(n-1)!$ cases, giving the approximation:

$$D_n = n! - n(n-1)!.$$

This quantity is clearly 0 and this happens because we have subtracted twice every permutation with at least 2 fixed points: in fact, we subtracted it when we considered the first and the second fixed point. Therefore, we have now to add permutations with at least two fixed points. These are obtained by choosing the two fixed points in all the $\binom{n}{2}$ possible ways and then permuting the $n-2$ remaining elements. Thus we have the new approximation:

$$D_n = n! - n(n-1)! + \binom{n}{2}(n-2)!.$$

In this way, however, we added twice permutations with at least three fixed points, which have to be subtracted again. We thus obtain:

$$D_n = n! - n(n-1)! + \binom{n}{2}(n-2)! - \binom{n}{3}(n-3)!.$$

We can now go on with the same method, which is called the *inclusion exclusion principle*, and eventually arrive to the final value:

$$\begin{aligned} D_n &= n! - n(n-1)! + \binom{n}{2}(n-2)! - \\ &\quad - \binom{n}{3}(n-3)! + \cdots = \\ &= \frac{n!}{0!} - \frac{n!}{1!} + \frac{n!}{2!} - \frac{n!}{3!} + \cdots = n!\sum_{k=0}^{n}\frac{(-1)^k}{k!}. \end{aligned}$$

This formula checks with the previously found values. We obtain the exponential generating function $\mathcal{G}(D_n/n!)$ by observing that the generic element in the sum is the coefficient $[t^n]e^{-t}$, and therefore by the theorem on the generating function for the partial sums we have:

$$\mathcal{G}\left(\frac{D_n}{n!}\right) = \frac{e^{-t}}{1-t}.$$

In order to find the asymptotic value for D_n, we observe that the radius of convergence of $1/(1-t)$ is 1, while e^{-t} converges for every value of t. By Bender's theorem we have:

$$\frac{D_n}{n!} \sim e^{-1} \quad \text{or} \quad D_n \sim \frac{n!}{e}.$$

This value is indeed a very good approximation for D_n, which can actually be computed as the integer nearest to $n!/e$.

Let us now see how Bender's theorem is applied to the exponential generating function of the ordered Bell numbers. We have shown that the dominating singularity is a pole at $t = \ln 2$ which has order 1:

$$\lim_{t\to\ln 2} \frac{1-t/\ln 2}{2-e^t} = \lim_{t\to\ln 2} \frac{-1/\ln 2}{-e^t} = \frac{1}{2\ln 2}.$$

At this point we have:

$$[t^n]\frac{1}{2-e^t} = [t^n]\frac{1}{1-t/\ln 2}\frac{1-t/\ln 2}{2-e^t} \sim$$
$$\sim \quad \left[\frac{1-t/\ln 2}{2-e^t} \;\middle|\; t=\ln 2\right][t^n]\frac{1}{1-t/\ln 2} =$$
$$= \quad \frac{1}{2}\frac{1}{(\ln 2)^{n+1}}$$

and we conclude with the very good approximation $\mathcal{O}_n \sim n!/(2(\ln 2)^{n+1})$.

Finally, we find the asymptotic approximation for the Bernoulli numbers. The following statement is very important when we have functions with several dominating singularity:

Principle: If $t_1, t_2, \ldots, t_k$ are all the dominating singularities of a function $f(t)$, then $[t^n]f(t)$ can be found by summing all the contributions obtained by independently considering the k singularities.

We already observed that $\pm 2\pi i$ are the two dominating singularities for the generating function of the Bernoulli numbers; they are both poles of order 1:

$$\lim_{t\to 2\pi i} \frac{t(1-t/2\pi i)}{e^t-1} = \lim_{t\to 2\pi i} \frac{1-t/\pi i}{e^t} = -1.$$
$$\lim_{t\to -2\pi i} \frac{t(1+t/2\pi i)}{e^t-1} = \lim_{t\to -2\pi i} \frac{1+t/\pi i}{e^t} = -1.$$

Therefore we have:

$$[t^n]\frac{t}{e^t-1} = [t^n]\frac{1}{1-t/2\pi i}\frac{t(1-t/2\pi i)}{e^t-1} \sim -\frac{1}{(2\pi i)^n}.$$

A similar result is obtained for the other pole; thus we have:

$$\frac{B_n}{n!} \sim -\frac{1}{(2\pi i)^n} - \frac{1}{(-2\pi i)^n}.$$

When n is odd, these two values are opposite in sign and the result is 0; this confirms that the Bernoulli numbers of odd index are 0, except for $n = 1$. When n is even, say $n = 2k$, we have $(2\pi i)^{2k} = (-2\pi i)^{2k} = (-1)^k(2\pi)^{2k}$; therefore:

$$B_{2k} \sim -\frac{2(-1)^k(2k)!}{(2\pi)^{2k}}.$$

This formula is a good approximation, also for small values of n, and shows that Bernoulli numbers become, in modulus, larger and larger as n increases.

7.5 Algebraic and logarithmic singularities

Let us consider the generating function for the Catalan numbers $f(t) = (1-\sqrt{1-4t})/(2t)$ and the corresponding power series $\widehat{f}(t) = 1 + t + 2t^2 + 5t^3 + 14t^4 + \cdots$. Our choice of the $-$ sign was motivated by the initial condition of the recurrence $C_{n+1} = \sum_{k=0}^{n} C_k C_{n-k}$ defining the Catalan numbers. This is due to the fact that, when the argument is a positive real number, we can choose the positive value as the result of a square root. In other words, we consider the *arithmetic square root* instead of the algebraic square root. This allows us to identify the power series $\widehat{f}(t)$ with the function $f(t)$, but when we pass to complex numbers this is no longer possible. Actually, in the complex field, a function containing a square root is a two-valued function, and there are two *branches* defined by the same expression. Only one of these two branches coincides with the function defined by the power series, which is obviously a one-valued function.

The points at which a square root becomes 0 are special points; in them the function is one-valued, but in every neighborhood the function is two-valued. For the smallest in modulus among these points, say t_0, we must have the following situation: for t such that $|t| < |t_0|$, $\widehat{f}(t)$ should coincide with a branch of $f(t)$, while for t such that $|t| > |t_0|$, $\widehat{f}(t)$ cannot converge. In fact, consider a $t \in \mathbb{R}$, $t > |t_0|$; the expression under the square root should be a negative real number and therefore $f(t) \in \mathbb{C}\setminus\mathbb{R}$; but $\widehat{f}(t)$ can only be a real number or $f(t)$ does not converge. Because we know that when $\widehat{f}(t)$ converges we must have $\widehat{f}(t) = f(t)$, we conclude that $\widehat{f}(t)$ cannot converge. This shows that t_0 is a singularity for $f(t)$.

Every kth root originates the same problem and the function is actually a k-valued function; all the values for which the argument of the root is 0 is a singularity, called an *algebraic singularity*. They can be treated by Darboux' method or, directly, by means of Bender's theorem, which relies on Newton's rule. Actually, we already used this method to find the asymptotic evaluation for the Motzkin numbers.

The same considerations hold when a function contains a logarithm. In fact, a logarithm is an infinite-valued function, because it is the inverse of the exponential, which, in the complex field $\mathbb{C}$, is a periodic function:

$$e^{t+2k\pi i} = e^t e^{2k\pi i} = e^t(\cos 2k\pi + i\sin 2k\pi) = e^t.$$

The period of e^t is therefore $2\pi i$ and $\ln t$ is actually $\ln t + 2k\pi i$, for $k \in \mathbb{Z}$. A point t_0 for which the argument of a logarithm is 0 is a singularity for the

corresponding function. In every neighborhood of t_0, the function has an infinite number of branches; this is the only fact distinguishing a *logarithmic singularity* from an algebraic one.

Let us suppose we have the sum:

$$S_n = 1 + \frac{2}{2} + \frac{4}{3} + \frac{8}{4} + \cdots + \frac{2^{n-1}}{n} = \frac{1}{2}\sum_{k=1}^{n}\frac{2^k}{k}$$

and we wish to compute an approximate value. The generating function is:

$$\mathcal{G}\left(\frac{1}{2}\sum_{k=1}^{n}\frac{2^k}{k}\right) = \frac{1}{2}\frac{1}{1-t}\ln\frac{1}{1-2t}.$$

There are two singularities: $t = 1$ is a pole, while $t = 1/2$ is a logarithmic singularity. Since the latter has smaller modulus, it is dominating and $R = 1/2$ is the radius of convergence of the function. By Bender's theorem we have:

$$\begin{aligned} S_n &= \frac{1}{2}[t^n]\frac{1}{1-t}\ln\frac{1}{1-2t} \sim \\ &\sim \frac{1}{2}\frac{1}{1-1/2}[t^n]\ln\frac{1}{1-2t} = \frac{2^n}{n}. \end{aligned}$$

This is not a very good approximation. In the next section we will see how it can be improved.

7.6 Subtracted singularities

The methods presented in the preceding sections only give the expression describing the general behavior of the coefficients f_n in the expansion $\widehat{f}(t) = \sum_{k=0}^{\infty} f_k t^k$, i.e., what is called the *principal value* for f_n. Sometimes, this behavior is only achieved for very large values of n, but for smaller values it is just a rough approximation of the true value. Because of that, we speak of "asymptotic evaluation" or "asymptotic approximation". When we need a true approximation, we should introduce some corrections, which slightly modify the general behavior and more accurately evaluate the true value of f_n.

Many times, the following observation solves the problem. Suppose we have found, by one of the previous methods, that a function $f(t)$ is such that $f(t) \sim A(1-\alpha t)^\gamma$, for some $A, \gamma \in \mathbb{R}$, or, more in general, $f(t) \sim g(t)$, for some function $g(t)$ of which we exactly know the coefficients g_n. For example, this is the case of $\ln(1/(1-\alpha t))$. Because $f_n \sim g_n$, the function $h(t) = f(t) - g(t)$ has coefficients h_n that grow more slowly than f_n; formally, since $O(f_n) = O(g_n)$, we must have $h_n = o(f_n)$. Therefore, the quantity $g_n + h_n$ is a better approximation to f_n than g_n.

When $f(t)$ has a pole t_0 of order m, the successive application of this method of *subtracted singularities* completely eliminates the singularity. Therefore, if a second singularity t_1 exists such that $|t_1| > |t_0|$, we can express the coefficients f_n in terms of t_1 as well. When $f(t)$ has a dominating algebraic singularity, this cannot be eliminated, but the method of subtracted singularities allows us to obtain corrections to the principal value. Formally, by the successive application of this method, we arrive to the following results. If t_0 is a dominating pole of order m and $\alpha = 1/t_0$, then we find the expansion:

$$f(t) = \frac{A_{-m}}{(1-\alpha t)^m} + \frac{A_{-m+1}}{(1-\alpha t)^{m-1}} + \frac{A_{-m+2}}{(1-\alpha t)^{m-2}} + \cdots.$$

If t_0 is a dominating algebraic singularity and $f(t) = h(t)(1-\alpha t)^{p/m}$, where $\alpha = 1/t_0$ and $h(t)$ has a radius of convergence larger than $|t_0|$, then we find the expansion:

$$\begin{aligned} f(t) &= A_p(1-\alpha t)^{p/m} + A_{p-1}(1-\alpha t)^{(p-1)/m} + \\ &\quad + A_{p-2}(1-\alpha t)^{(p-2)/m} + \cdots. \end{aligned}$$

Newton's rule can obviously be used to pass from these expansions to the asymptotic value of f_n.

The same method of subtracted singularities can be used for a logarithmic singularity. Let us consider as an example the sum $S_n = \sum_{k=1}^{n} 2^{k-1}/k$, introduced in the previous section. We found the principal value $S_n \sim 2^n/n$ by studying the generating function:

$$\frac{1}{2}\frac{1}{1-t}\ln\frac{1}{1-2t} \sim \ln\frac{1}{1-2t}.$$

Let us therefore consider the new function:

$$\begin{aligned} h(t) &= \frac{1}{2}\frac{1}{1-t}\ln\frac{1}{1-2t} - \ln\frac{1}{1-2t} = \\ &= -\frac{1-2t}{2(1-t)}\ln\frac{1}{1-2t}. \end{aligned}$$

The generic term h_n should be significantly less than f_n; the factor $(1-2t)$ actually reduces the order of growth of the logarithm:

$$\begin{aligned} &-[t^n]\frac{1-2t}{2(1-t)}\ln\frac{1}{1-2t} = \\ &= -\frac{1}{2(1-1/2)}[t^n](1-2t)\ln\frac{1}{1-2t} = \\ &= -\left(\frac{2^n}{n} - 2\frac{2^{n-1}}{n-1}\right) = \frac{2^n}{n(n-1)}. \end{aligned}$$

Therefore, a better approximation for S_n is:

$$S_n = \frac{2^n}{n} + \frac{2^n}{n(n-1)} = \frac{2^n}{n-1}.$$

The reader can easily verify that this correction greatly reduces the error in the evaluation of S_n.

A further correction can now be obtained by considering:

$$\begin{aligned} k(t) &= \frac{1}{2}\frac{1}{1-t}\ln\frac{1}{1-2t} - \\ &\quad -\ln\frac{1}{1-2t} + (1-2t)\ln\frac{1}{1-2t} = \\ &= \frac{(1-2t)^2}{2(1-t)}\ln\frac{1}{1-2t} \end{aligned}$$

which gives:

$$\begin{aligned} k_n &= \frac{1}{2(1-1/2)}[t^n](1-2t)^2\ln\frac{1}{1-2t} = \\ &= \frac{2^n}{n} - 4\frac{2^{n-1}}{n-1} + 4\frac{2^{n-2}}{n-2} = \frac{2^{n+1}}{n(n-1)(n-2)}. \end{aligned}$$

This correction is still smaller, and we can write:

$$S_n \sim \frac{2^n}{n-1}\left(1+\frac{2}{n(n-2)}\right).$$

In general, we can obtain the same results if we expand the function $h(t)$ in $f(t) = g(t)h(t)$, $h(t)$ with a radius of convergence larger than that of $f(t)$, around the dominating singularity. This is done in the following way:

$$\begin{aligned} &\frac{1}{2(1-t)} = \frac{1}{1+(1-2t)} = \\ &= 1-(1-2t)+(1-2t)^2-(1-2t)^3+\cdots. \end{aligned}$$

This implies:

$$\begin{aligned} \frac{1}{2(1-t)}\ln\frac{1}{1-2t} &= \ln\frac{1}{1-2t} - \\ &-(1-2t)\ln\frac{1}{1-2t} + (1-2t)^2\ln\frac{1}{1-2t} - \cdots \end{aligned}$$

and the result is the same as the one previously obtained by the method of subtracted singularities.

7.7 The asymptotic behavior of a trinomial square root

In many problems we arrive to a generating function of the form:

$$f(t) = \frac{p(t) - \sqrt{(1-\alpha t)(1-\beta t)}}{rt^m}$$

or:

$$g(t) = \frac{q(t)}{\sqrt{(1-\alpha t)(1-\beta t)}}.$$

In the former case, $p(t)$ is a correcting polynomial, which has no effect on f_n, for n sufficiently large, and therefore we have:

$$f_n = -\frac{1}{r}[t^{n+m}]\sqrt{(1-\alpha t)(1-\beta t)}$$

where m is a small integer. In the second case, g_n is the sum of various terms, as many as there are terms in the polynomial $q(t)$, each one of the form:

$$q_k[t^{n-k}]\frac{1}{\sqrt{(1-\alpha t)(1-\beta t)}}.$$

It is therefore interesting to compute, once and for all, the asymptotic value $[t^n]((1-\alpha t)(1-\beta t))^s$, where $s = 1/2$ or $s = -1/2$.

Let us suppose that $|\alpha| > |\beta|$, since the case $\alpha = \beta$ has no interest and the case $\alpha = -\beta$ should be approached in another way. This hypothesis means that $t = 1/\alpha$ is the radius of convergence of the function and we can develop everything around this singularity. In most combinatorial problems we have $\alpha > 0$, because the coefficients of $f(t)$ are positive numbers, but this is not a limiting factor.

Let us consider $s = 1/2$; in this case, a minus sign should precede the square root. The evaluation is shown in Table 7.1. The formula so obtained can be considered sufficient for obtaining both the asymptotic evaluation of f_n and a suitable numerical approximation. However, we can use the following developments:

$$\begin{aligned} \binom{2n}{n} &= \frac{4^n}{\sqrt{\pi n}}\left(1-\frac{1}{8n}+\frac{1}{128n^2}+O\left(\frac{1}{n^3}\right)\right) \\ \frac{1}{2n-1} &= \frac{1}{2n}\left(1+\frac{1}{2n}+\frac{1}{4n^2}+O\left(\frac{1}{n^3}\right)\right) \\ \frac{1}{2n-3} &= \frac{1}{2n}\left(1+\frac{3}{2n}+O\left(\frac{1}{n^2}\right)\right) \end{aligned}$$

and get:

$$\begin{aligned} f_n = \sqrt{\frac{\alpha-\beta}{\alpha}}\frac{\alpha^n}{2n\sqrt{\pi n}}\Bigg(1-\frac{6+3(\alpha-\beta)}{8(\alpha-\beta)n}+\frac{25}{128n^2}+ \\ +\frac{9}{8(\alpha-\beta)n^2}-\frac{9\alpha\beta+51\beta^2}{32(\alpha-\beta)^2n^2}+O\left(\frac{1}{n^3}\right)\Bigg). \end{aligned}$$

The reader is invited to find a similar formula for the case $s = 1/2$.

7.8 Hayman's method

The method for coefficient evaluation which uses the function singularities (Darboux' method), can be improved and made more accurate, as we have seen, by the technique of "subtracted singularities". Unfortunately, these methods become useless when the function $f(t)$ has no singularity (*entire* functions) or when the dominating singularity is essential. In fact, in the former case we do not have any singularity to operate on, and in the latter the development around the singularity gives rise to a series with an infinite number of terms of negative degree.

$$
\begin{gathered}
[t^n]-(1-\alpha t)^{1/2}(1-\beta t)^{1/2}=[t^n]-(1-\alpha t)^{1/2}\left(\tfrac{\alpha-\beta}{\alpha}+\tfrac{\beta}{\alpha}(1-\alpha t)\right)^{1/2}= \\
=-\sqrt{\tfrac{\alpha-\beta}{\alpha}}[t^n](1-\alpha t)^{1/2}\left(1+\tfrac{\beta}{\alpha-\beta}(1-\alpha t)\right)^{1/2}= \\
=-\sqrt{\tfrac{\alpha-\beta}{\alpha}}[t^n](1-\alpha t)^{1/2}\left(1+\tfrac{\beta}{2(\alpha-\beta)}(1-\alpha t)-\tfrac{\beta^2}{8(\alpha-\beta)^2}(1-\alpha t)^2+\cdots\right)= \\
=-\sqrt{\tfrac{\alpha-\beta}{\alpha}}[t^n]\left((1-\alpha t)^{1/2}+\tfrac{\beta}{2(\alpha-\beta)}(1-\alpha t)^{3/2}-\tfrac{\beta^2}{8(\alpha-\beta)^2}(1-\alpha t)^{5/2}+\cdots\right)= \\
=-\sqrt{\tfrac{\alpha-\beta}{\alpha}}\left(\tbinom{1/2}{n}(-\alpha)^n+\tfrac{\beta}{2(\alpha-\beta)}\tbinom{3/2}{n}(-\alpha)^n-\tfrac{\beta^2}{8(\alpha-\beta)^2}\tbinom{5/2}{n}(-\alpha)^n+\cdots\right)= \\
=-\sqrt{\tfrac{\alpha-\beta}{\alpha}}\tfrac{(-1)^{n-1}}{4^n(2n-1)}\tbinom{2n}{n}(-\alpha)^n\left(1-\tfrac{\beta}{2(\alpha-\beta)}\tfrac{3}{2n-3}-\tfrac{\beta^2}{8(\alpha-\beta)^2}\tfrac{15}{(2n-3)(2n-5)}+\cdots\right)= \\
=\sqrt{\tfrac{\alpha-\beta}{\alpha}}\tfrac{\alpha^n}{4^n(2n-1)}\tbinom{2n}{n}\left(1-\tfrac{3\beta}{2(\alpha-\beta)(2n-3)}-\tfrac{15\beta^2}{8(\alpha-\beta)^2(2n-3)(2n-5)}+O\left(\tfrac{1}{n^3}\right)\right).
\end{gathered}
$$

Table 7.1: The case $s = 1/2$

In these cases, the only method seems to be the Cauchy theorem, which allows us to evaluate $[t^n]f(t)$ by means of an integral:

$$f_n = \frac{1}{2\pi i}\int_\gamma \frac{f(t)}{t^{n+1}}dt$$

where γ is a suitable path enclosing the origin. We do not intend to develop this method here, but we'll limit ourselves to sketch a method, derived from Cauchy theorem, which allows us to find an asymptotic evaluation for f_n in many practical situations. The method can be implemented on a computer in the following sense: given a function $f(t)$, in an algorithmic way we can check whether $f(t)$ belongs to the class of functions for which the method is applicable (the class of "H-admissible" functions) and, if that is the case, we can evaluate the principal value of the asymptotic estimate for f_n. The system $\Lambda\Upsilon\Omega$, by Flajolet, Salvy and Zimmermann, realizes this method. The development of the method was mainly performed by Hayman and therefore it is known as *Hayman's method*; this also justifies the use of the letter H in the definition of H-admissibility.

A function is called *H-admissible* if and only if it belongs to one of the following classes or can be obtained, in a finite number of steps according to the following rules, from other H-admissible functions:

1. if $f(t)$ and $g(t)$ are H-admissible functions and $p(t)$ is a polynomial with real coefficients and positive leading term, then:

$$\exp(f(t)) \quad f(t)+g(t) \quad f(t)+p(t)$$

$$p(f(t)) \quad p(t)f(t)$$

   are all H-admissible functions;

2. if $p(t)$ is a polynomial with positive coefficients and not of the form $p(t^k)$ for $k > 1$, then the function $\exp(p(t))$ is H-admissible;

3. if α, β are positive real numbers and γ, δ are real numbers, then the function:

$$f(t) = \exp\left(\frac{\beta}{(1-t)^\alpha}\left(\frac{1}{t}\ln\frac{1}{(1-t)}\right)^\gamma \times \right.$$
$$\left. \times \left(\frac{2}{t}\ln\left(\frac{1}{t}\ln\frac{1}{(1-t)}\right)\right)^\delta\right)$$

   is H-admissible.

For example, the following functions are all H-admissible:

$$e^t \quad \exp\left(t+\frac{t^2}{2}\right) \quad \exp\left(\frac{t}{1-t}\right)$$
$$\exp\left(\frac{1}{t(1-t)^2}\ln\frac{1}{1-t}\right).$$

In particular, for the third function we have:

$$\exp\left(\frac{t}{1-t}\right) = \exp\left(\frac{1}{1-t}-1\right) = \frac{1}{e}\exp\left(\frac{1}{1-t}\right)$$

and naturally a constant does not influence the H-admissibility of a function. In this example we have $\alpha = \beta = 1$ and $\gamma = \delta = 0$.

For H-admissible functions, the following result holds:

Theorem 7.8.1 *Let $f(t)$ be an H-admissible function; then:*

$$f_n = [t^n]f(t) \sim \frac{f(r)}{r^n\sqrt{2\pi b(r)}} \qquad \text{as} \quad n\to\infty$$

where $r = r(n)$ is the least positive solution of the equation $tf'(t)/f(t) = n$ and $b(t)$ is the function:

$$b(t) = t\frac{d}{dt}\left(t\frac{f'(t)}{f(t)}\right).$$

As we said before, the proof of this theorem is based on Cauchy's theorem and is beyond the scope of these notes. Instead, let us show some examples to clarify the application of Hayman's method.

7.9 Examples of Hayman's Theorem

The first example can be easily verified. Let $f(t) = e^t$ be the exponential function, so that we know $f_n = 1/n!$. For applying Hayman's theorem, we have to solve the equation $te^t/e^t = n$, which gives $r = n$. The function $b(t)$ is simply t and therefore we have:

$$[t^n]e^t = \frac{e^n}{n^n\sqrt{2\pi n}}$$

and in this formula we immediately recognize Stirling approximation for factorials.

Examples become early rather complex and require a large amount of computations. Let us consider the following sum:

$$\begin{aligned}
\sum_{k=0}^{n-1}\binom{n-1}{k}\frac{1}{(k+1)!} &= \sum_{k=0}^{n}\binom{n-1}{k-1}\frac{1}{k!} = \\
&= \sum_{k=0}^{n}\left(\binom{n}{k}-\binom{n-1}{k}\right)\frac{1}{k!} = \\
&= \sum_{k=0}^{n}\binom{n}{k}\frac{1}{k!} - \sum_{k=0}^{n-1}\binom{n-1}{k}\frac{1}{k!} = \\
&= [t^n]\frac{1}{1-t}\exp\left(\frac{t}{1-t}\right) - \\
&\quad - [t^{n-1}]\frac{1}{1-t}\exp\left(\frac{t}{1-t}\right) = \\
&= [t^n]\frac{1-t}{1-t}\exp\left(\frac{t}{1-t}\right) = [t^n]\exp\left(\frac{t}{1-t}\right).
\end{aligned}$$

We have already seen that this function is H-admissible, and therefore we can try to evaluate the asymptotic behavior of the sum. Let us define the function $g(t) = tf'(t)/f(t)$, which in the present case is:

$$g(t) = \frac{\frac{t}{(1-t)^2}\exp\left(\frac{t}{1-t}\right)}{\exp\left(\frac{t}{1-t}\right)} = \frac{t}{(1-t)^2}.$$

The value of r is therefore given by the minimal positive solution of:

$$\frac{t}{(1-t)^2} = n \quad \text{or} \quad nt^2 - (2n+1)t + n = 0.$$

Because $\Delta = 4n^2 + 4n + 1 - 4n^2 = 4n + 1$, we have the two solutions:

$$r = \frac{2n+1 \pm \sqrt{4n+1}}{2n}$$

and we must accept the one with the '$-$' sign, which is positive and less than the other. It is surely positive, because $\sqrt{4n+1} < 2\sqrt{n} + 1 < 2n$, for every $n > 1$.

As n grows, we can also give an asymptotic approximation of r, by developing the expression inside the square root:

$$\begin{aligned}
\sqrt{4n+1} &= 2\sqrt{n}\sqrt{1+\frac{1}{4n}} = \\
&= 2\sqrt{n}\left(1+\frac{1}{8n}+O\left(\frac{1}{n^2}\right)\right).
\end{aligned}$$

From this formula we immediately obtain:

$$r = 1 - \frac{1}{\sqrt{n}} + \frac{1}{2n} - \frac{1}{8n\sqrt{n}} + O\left(\frac{1}{n^2}\right)$$

which will be used in the next approximations. First we compute an approximation for $f(r)$, that is $\exp(r/(1-r))$. Since:

$$\begin{aligned}
1 - r &= \frac{1}{\sqrt{n}} - \frac{1}{2n} + \frac{1}{8n\sqrt{n}} + O\left(\frac{1}{n^2}\right) = \\
&= \frac{1}{\sqrt{n}}\left(1 - \frac{1}{2\sqrt{n}} + \frac{1}{8n} + O\left(\frac{1}{n\sqrt{n}}\right)\right)
\end{aligned}$$

we immediately obtain:

$$\begin{aligned}
\frac{r}{1-r} &= \left(1 - \frac{1}{\sqrt{n}} + \frac{1}{2n} + O\left(\frac{1}{n\sqrt{n}}\right)\right) \times \\
&\quad \times\sqrt{n}\left(1 + \frac{1}{2\sqrt{n}} + \frac{1}{8n} + O\left(\frac{1}{n\sqrt{n}}\right)\right) = \\
&= \sqrt{n}\left(1 - \frac{1}{2\sqrt{n}} + \frac{1}{8n} + O\left(\frac{1}{n\sqrt{n}}\right)\right) = \\
&= \sqrt{n} - \frac{1}{2} + \frac{1}{8\sqrt{n}} + O\left(\frac{1}{n}\right).
\end{aligned}$$

Finally, the exponential gives:

$$\begin{aligned}
\exp\left(\frac{r}{1-r}\right) &= \frac{e^{\sqrt{n}}}{\sqrt{e}}\exp\left(\frac{1}{8\sqrt{n}} + O\left(\frac{1}{n}\right)\right) = \\
&= \frac{e^{\sqrt{n}}}{\sqrt{e}}\left(1 + \frac{1}{8\sqrt{n}} + O\left(\frac{1}{n}\right)\right).
\end{aligned}$$

Because Hayman's method only gives the principal value of the result, the correction can be ignored (it can be not precise) and we get:

$$\exp\left(\frac{r}{1-r}\right) = \frac{e^{\sqrt{n}}}{\sqrt{e}}.$$

The second part we have to develop is $1/r^n$, which can be computed when we write it as $\exp(n \ln 1/r)$, that is:

$$\begin{aligned}
\frac{1}{r^n} &= \exp\left(n\ln\left(1 + \frac{1}{\sqrt{n}} + \frac{1}{2n} + O\left(\frac{1}{n\sqrt{n}}\right)\right)\right) = \\
&= \exp\left(n\left(\frac{1}{\sqrt{n}} + \frac{1}{2n} + O\left(\frac{1}{n\sqrt{n}}\right) - \right.\right. \\
&\quad \left.\left. -\frac{1}{2}\left(\frac{1}{\sqrt{n}} + O\left(\frac{1}{n\sqrt{n}}\right)\right)^2 + O\left(\frac{1}{n\sqrt{n}}\right)\right)\right)
\end{aligned}$$

$$= \exp\left(n\left(\frac{1}{\sqrt{n}}+\frac{1}{2n}-\frac{1}{2n}+O\left(\frac{1}{n\sqrt{n}}\right)\right)\right) =$$
$$= \exp\left(\sqrt{n}+O\left(\frac{1}{\sqrt{n}}\right)\right) \sim e^{\sqrt{n}}.$$

Again, the correction is ignored and we only consider the principal value. We now observe that $f(r)/r^n \sim e^{2\sqrt{n}}/\sqrt{e}$.

Only $b(r)$ remains to be computed; we have $b(t) = tg'(t)$ where $g(t)$ is as above, and therefore we have:

$$b(t) = t\frac{d}{dt}\frac{t}{(1-t)^2} = \frac{t(1+t)}{(1-t)^3}.$$

This quantity can be computed a piece at a time. First:

$$r(1+r) = \left(1-\frac{1}{\sqrt{n}}+\frac{1}{2n}+O\left(\frac{1}{n\sqrt{n}}\right)\right)\times$$
$$\times\left(2-\frac{1}{\sqrt{n}}+\frac{1}{2n}+O\left(\frac{1}{n\sqrt{n}}\right)\right) =$$
$$= 2-\frac{3}{\sqrt{n}}+\frac{5}{2n}+O\left(\frac{1}{n\sqrt{n}}\right) =$$
$$= 2\left(1-\frac{3}{2\sqrt{n}}+\frac{5}{4n}+O\left(\frac{1}{n\sqrt{n}}\right)\right).$$

By using the expression already found for $(1-r)$ we then have:

$$(1-r)^3 = \frac{1}{n\sqrt{n}}\left(1-\frac{1}{2\sqrt{n}}+\frac{1}{8n}+O\left(\frac{1}{n\sqrt{n}}\right)\right)^3 =$$
$$= \frac{1}{n\sqrt{n}}\left(1-\frac{1}{\sqrt{n}}+\frac{1}{2n}+O\left(\frac{1}{n\sqrt{n}}\right)\right)\times$$
$$\times\left(1-\frac{1}{2\sqrt{n}}+\frac{1}{8n}+O\left(\frac{1}{n\sqrt{n}}\right)\right) =$$
$$= \frac{1}{n\sqrt{n}}\left(1-\frac{3}{2\sqrt{n}}+\frac{9}{8n}+O\left(\frac{1}{n\sqrt{n}}\right)\right).$$

By inverting this quantity, we eventually get:

$$b(r) = \frac{r(1+r)}{(1-r)^3} =$$
$$= 2n\sqrt{n}\left(1+\frac{3}{2\sqrt{n}}+\frac{9}{8n}+O\left(\frac{1}{n\sqrt{n}}\right)\right)\times$$
$$\times\left(1-\frac{3}{2\sqrt{n}}+\frac{5}{4n}+O\left(\frac{1}{n\sqrt{n}}\right)\right) =$$
$$= 2n\sqrt{n}\left(1+\frac{1}{8n}+O\left(\frac{1}{n\sqrt{n}}\right)\right).$$

The principal value is $2n\sqrt{n}$ and therefore:

$$\sqrt{2\pi b(r)} = \sqrt{4\pi n\sqrt{n}} = 2\sqrt{\pi}n^{3/4};$$

the final result is:

$$f_n = [t^n]\exp\left(\frac{t}{1-t}\right) \sim \frac{e^{2\sqrt{n}}}{2\sqrt{\pi e}n^{3/4}}.$$

It is a simple matter to execute the original sum $\sum_{k=0}^{n-1}\binom{n-1}{k}/(k+1)!$ by using a personal computer for, say, $n = 100, 200, 300$. The results obtained can be compared with the evaluation of the previous formula. We can thus verify that the relative error decreases as n increases.

Chapter 8

Bibliography

The birth of Computer Science and the need of analyzing the behavior of algorithms and data structures have given a strong twirl to Combinatorial Analysis, and to the mathematical methods used to study combinatorial objects. So, near to the traditional literature on Combinatorics, a number of books and papers have been produced, relating Computer Science and the methods of Combinatorial Analysis. The first author who systematically worked in this direction was surely Donald Knuth, who published in 1968 the first volume of his monumental "*The Art of Computer Programming*", the first part of which is dedicated to the mathematical methods used in Combinatorial Analysis. Without this basic knowledge, there is little hope to understand the developments of the analysis of algorithms and data structures:

Donald E. Knuth: *The Art of Computer Programming: Fundamental Algorithms*, Vol. I, Addison-Wesley (1968).

Many additional concepts and techniques are also contained in the third volume:

Donald E. Knuth: *The Art of Computer Programming: Sorting and Searching*, Vol. III, Addison-Wesley (1973).

Numerical and probabilistic developments are to be found in the central volume:

Donald E. Knuth: *The Art of Computer Programming: Numerical Algorithms*, Vol. II, Addison-Wesley (1973)

A concise exposition of several important techniques is given in:

Daniel H. Greene, Donald E. Knuth: *Mathematics for the Analysis of Algorithms*, Birkhäuser (1981).

The most systematic work in the field is perhaps the following text, very clear and very comprehensive. It deals, in an elementary but rigorous way, with the main topics of Combinatorial Analysis, with an eye to Computer Science. Many exercise (with solutions) are proposed, and the reader is never left alone in front of the many-faced problems he or she is encouraged to tackle.

Ronald L. Graham, Donald E. Knuth, Oren Patashnik: *Concrete Mathematics*, Addison-Wesley (1989).

Many texts on Combinatorial Analysis are worth of being considered, because they contain information on general concepts, both from a combinatorial and a mathematical point of view:

William Feller: *An Introduction to Probability Theory and Its Applications*, Wiley (1950) - (1957) - (1968).

John Riordan: *An Introduction to Combinatorial Analysis*, Wiley (1953) (1958).

Louis Comtet: *Advanced Combinatorics*, Reidel (1974).

Ian P. Goulden, David M. Jackson: Combinatorial Enumeration, Dover Publ. (2004).

Richard P. Stanley: *Enumerative Combinatorics*, Vol. I, Cambridge Univ. Press (1986) (2000).

Richard P. Stanley: *Enumerative Combinatorics*, Vol. II, Cambridge Univ. Press !997) (2001).

Robert Sedgewick, Philippe Flajolet: *An Introduction to the Analysis of Algorithms*, Addison-Wesley (1995).

* * *

Chapter 1: Introduction

The concepts relating to the problem of searching can be found in any text on Algorithms and Data Structures, in particular Volume III of the quoted "*The Art of Computer Programming*". Volume I can be consulted for Landau notation, even if it is common in Mathematics. For the mathematical concepts of Γ and ψ functions, the reader is referred to:

Milton Abramowitz, Irene A. Stegun: *Handbook of Mathematical Functions*, Dover Publ. (1972).

* * *

Chapter 2: Special numbers

The quoted book by Graham, Knuth and Patashnik is appropriate. Anyhow, the quantity we consider are common to all parts of Mathematics, in particular of Combinatorial Analysis. The ζ function and the Bernoulli numbers are also covered by the book of Abramowitz and Stegun. Stanley dedicates to Catalan numbers a special survey in his second volume. Probably, they are the most frequently used quantity in Combinatorics, just after binomial coefficients and before Fibonacci numbers. Stirling numbers are very important in Numerical Analysis, but here we are more interested in their combinatorial meaning. Harmonic numbers are the "discrete" version of logarithms, and from this fact their relevance in Combinatorics and in the Analysis of Algorithms originates.

Sequences studied in Combinatorial Analysis have been collected and annotated by Sloane. The resulting book (and the corresponding Internet site) is one of the most important reference point:

N. J. A. Sloane, Simon Plouffe: *The Encyclopedia of Integer Sequences* Academic Press (1995). Available at:
http://www.research.att.com/ njas/sequences/.

* * *

Chapter 3: Formal Power Series

Formal power series have a long tradition; the reader can find their algebraic foundation in the book:

Peter Henrici: *Applied and Computational Complex Analysis*, Wiley (1974) Vol. I; (1977) Vol. II; (1986) Vol. III.

Coefficient extraction is central in our approach; a formalization can be found in:

Donatella Merlini, Renzo Sprugnol, M. Cecilia Verri: The method of coefficients, *The American Mathematical Monthly* (to appear).

The Lagrange Inversion Formula can be found in most of the quoted texts, in particular in Stanley and Henrici. Nowadays, almost every part of Mathematics is available by means of several Computer Algebra Systems, implementing on a computer the ideas described in this chapter, and many, many others. Maple and Mathematica are among the most used systems; other software is available, free or not. These systems have become an essential tool for developing any new aspect of Mathematics.

* * *

Chapter 4: Generating functions

In our present approach the concept of an (ordinary) generating function is essential, and this justifies our insistence on the two operators "generating function" and "coefficient of". Since their introduction in Mathematics, due to de Moivre in the XVIII Century, generating functions have been a controversial concept. Now they are accepted almost universally, and their theory has been developed in many of the quoted texts. In particular:

Herbert S. Wilf: *Generatingfunctionology*, Academic Press (1990).

A practical device has been realized in Maple:

Bruno Salvy, Paul Zimmermann: GFUN: a Maple package for the manipulation of generating and holonomic functions in one variable, INRIA, Rapport Technique N. 143 (1992).

The number of applications of generating functions is almost infinite, so we limit our considerations to some classical cases relative to Computer Science. Sometimes, we intentionally complicate proofs in order to stress the use of generating function as a unifying approach. So, our proofs should be compared with those given (for the same problems) by Knuth or Flajolet and Sedgewick.

* * *

Chapter 5: Riordan arrays

Riordan arrays are part of the general *method of coefficients* and are particularly important in proving combinatorial identities and generating function transformations. They were introduced by:

Louis W. Shapiro, Seyoum Getu, Wen-Jin Woan, Leon C. Woodson: The Riordan group, *Discrete Applied Mathematics*, 34 (1991) 229 – 239.

Their practical relevance was noted in:

Renzo Sprugnoli: Riordan arrays and combinatorial sums, *Discrete Mathematics*, 132 (1992) 267 – 290.

The theory was further developed in:

Donatella Merlini, Douglas G. Rogers, Renzo Sprugnoli, M. Cecilia Verri: On some alternative characterizations of Riordan arrays, *Canadian Journal of Mathematics* 49 (1997) 301 – 320.

Riordan arrays are strictly related to "convolution matrices" and to "Umbral calculus", although, rather strangely, nobody seems to have noticed the connections of these concepts and combinatorial sums:

Steve Roman: *The Umbral Calculus* Academic Press (1984).

Donald E. Knuth: Convolution polynomials, *The Methematica Journal*, 2 (1991) 67 – 78.

Collections of combinatorial identities are:

Henry W. Gould: *Combinatorial Identities. A Standardized Set of Tables Listing 500 Binomial Coefficient Summations* West Virginia University (1972).

Josef Kaucky: *Combinatorial Identities* Veda, Bratislava (1975).

Other methods to prove combinatorial identities are important in Combinatorial Analysis. We quote:

John Riordan: *Combinatorial Identities*, Wiley (1968).

G. P. Egorychev: Integral Representation and the Computation of Combinatorial Sums American Math. Society Translations, Vol. 59 (1984).

Doron Zeilberger: A holonomic systems approach to special functions identities, *Journal of Computational and Applied Mathematics* 32 (1990), 321 – 368.

Doron Zeilberger: A fast algorithm for proving terminating hypergeometric identities, *Discrete Mathematics* 80 (1990), 207 – 211.

M. Petkovšek, Herbert S. Wilf, Doron Zeilberger: $A = B$, A. K. Peters (1996).

In particular, the method of Wilf and Zeilberger completely solves the problem of combinatorial identities "with hypergeometric terms". This means that we can algorithmically establish whether an identity is true or false, provided the two members of the identity: $\sum_k L(k,n) = R(n)$ have a special form:

$$\frac{L(k+1,n)}{L(k,n)} \quad \frac{L(k,n+1)}{L(k,n)}$$

$$\frac{R(n+1)}{R(n)}$$

are all rational functions in n and k. This actually means that $L(k,n)$ and $R(n)$ are composed of factorial, powers and binomial coefficients. In this sense, Riordan arrays are less powerfy+ul, but can be used also when non-hypergeometric terms are not involved, as for examples in the case of harmonic and Stirling numbers of both kind.

* * *

Chapter 6: Formal Methods

In this chapter we have considered two important methods: the symbolic method for deducing counting generating functions from the syntactic definition of combinatorial objects, and the method of "operators" for obtaining combinatorial identities from relations between transformations of sequences defined by operators.

The symbolic method was started by Schützenberger and Viennot, who devised a technique to automatically generate counting generating functions from a context-free non-ambiguous grammar. When the grammar defines a class of combinatorial objects, this method gives a direct way to obtain monovariate or multivariate generating functions, which allow to solve many problems relative to the given objects. Since context-free languages only define algebraic generating functions (the subset of regular grammars is limited to rational

functions), the method is not very general, but is very effective whenever it can be applied. The method was extended by Flajolet to some classes of exponential generating functions and implemented in Maple.

Marco Schützenberger: Context-free languages and pushdown automata, *Information and Control* 6 (1963) 246 – 264

Maylis Delest, Xavier Viennot: Algebraic languages and polyominoes enumeration, *X Colloquium on Automata, Languages and Programming - Lecture Notes in Computer Science* (1983) 173 – 181.

Philippe Flajolet: Symbolic enumerative combinatorics and complex asymptotic analysis, *Algorithms Seminar*, (2001). Available at: http://algo.inria.fr/seminars/sem00-01/flajolet.html

The method of operators is very old and was developed in the XIX Century by English mathematicians, especially George Boole. A classical book in this direction is:

Charles Jordan: *Calculus of Finite Differences*, Chelsea Publ. (1965).

Actually, the method is used in Numerical Analysis, but it has a clear connection with Combinatorial Analysis, as our numerous examples show. The important concepts of indefinite and definite summations are used by Wilf and Zeilberger in the quoted texts. The Euler-McLaurin summation formula is the first connection between finite methods (considered up to this moment) and asymptotics.

* * *

Chapter 7: Asymptotics

The methods treated in the previous chapters are "exact", in the sense that every time they give the solution to a problem, this solution is a precise formula. This, however, is not always possible, and many times we are not able to find a solution of this kind. In these cases, we would also be content with an approximate solution, provided we can give an upper bound to the error committed. The purpose of asymptotic methods is just that.

The natural settings for these problems are Complex Analysis and the theory of series. We have used a rather descriptive approach, limiting our considerations to elementary cases. These situations are covered by the quoted text, especially Knuth, Wilf and Henrici. The method of Heyman is based on the paper:

Micha Hofri: *Probabilistic Analysis of Algorithms*, Springer (1987).

Index