Enumeration of Binary Strings and Applications to Compositions and Partitions

Maleka Raphadu(1407653)

A dissertation submitted to the Faculty of Science, University of the Witwatersrand, Johannesburg, in fulfilment of the requirements for the degree of Master of Science.

2023

DECLARATION

I **Maleka Raphadu**, declare that this dissertation is my original work and that I have properly cited and acknowledged any work borrowed from any other sources. It is been submitted to the University of the Witwatersrand, Johannesburg for consideration of the Master of Science degree. It has never been submitted for any other university's degree or examination.

Maleka Raphadu 03 October 2023

Abstract

In this dissertation, we first introduce binary strings and give a historical background. Then we discuss some techniques for enumerating restricted sets of binary strings with several examples. We employ mainly the symbolic method and recursive techniques, among others, to obtain our results. A chapter is devoted to a discussion of some published case studies on bit string enumerations which are relevant to our project. Then we consider how the study of binary strings may facilitate the enumeration of selected classes of compositions and integer partitions.

Contents

Chapter 1

Introduction

1.1 Introduction

A binary string or bit string is a sequence of 0's and 1's. A "bit" is a single binary digit of either 0 or 1. More formally we say that a binary string is a word over the alphabet $\{0,1\}$. A binary string of length n is any member of the set $\{0,1\}^n$, where $\{0,1\}^n$ denotes a Cartesian product with n terms:

$$\{0,1\}^n := \underbrace{\{0,1\} \times \{0,1\} \times \cdots \{0,1\}}_{\text{n times}}.$$

Example 1.1.1. We have that 0101001, 110100, 1 and 0 are examples of finite strings with lengths 7, 6, 1 and 1 respectively, and $010101\cdots$ is an infinite binary string.

The objective of this project is to study different methods of enumerating binary strings. We will also consider some applications of bit strings to the enumeration of integer partitions and compositions.

We first define an important enumeration tool - the generating function - which is associated with a given integer sequence.

Definition 1.1.1. A generating function of a sequence $\{a_n\}_{n=0}^{\infty}$ is a formal power series defined by

$$A(x) = \sum_{n=0}^{\infty} a_n x^n. \tag{1.1}$$

Example 1.1.2. Suppose $a_n = 1$ for every n. Then

$$A(x) = \sum_{n=0}^{\infty} x^n = 1 + x + x^2 + x^3 + x^4 + \cdots \tag{1.2}$$

Using the geometric series formula we can write (1.2) as

$$\sum_{n=0}^{\infty} x^n = \frac{1}{1-x},\ |x| < 1.$$

So the generating function of the sequence $\{1\}_{n\geq 0}$ is

$$A(x) = \frac{1}{1-x}.$$

We deduce from the foregoing result that a generating function of the sequence $\{2^n\}_{n\geq 0}$ is given by

$$\sum_{n=1}^{\infty} 2^{n-1}x^n = \frac{x}{1-2x} = x + 2x^2 + 4x^3 + 8x^4 + 16x^5 ... \tag{1.3}$$

or equivalently,

$$1 + \sum_{n=1}^{\infty} 2^{n-1}x^n = \frac{1-x}{1-2x} = 1 + x + 2x^2 + 4x^3 + 8x^4 + 16x^5 ... \tag{1.4}$$

Finally we give an illustration using the famous sequence of Fibonacci numbers:

$$1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, \ldots$$

Recall that the explicit formula for the nth Fibonacci number is

$$F_n = \frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^n - \frac{1}{\sqrt{5}}\left(\frac{1-\sqrt{5}}{2}\right)^n.$$

The nth Fibonacci number is also defined recursively by

$$F_n = F_{n-1} + F_{n-2} \qquad \text{with } F_0 = 0, F_1 = 1 \text{ for } n \geq 2. \tag{1.5}$$

The generating function is given by

$$\sum_{n=0}^{\infty} F_n x^n = \frac{x}{1-x-x^2} = x + x^2 + 2x^3 + 3x^4 + 5x^5 + \cdots + F_n x^n + \cdots , \qquad (1.6)$$

where F_n is given by $F_n = F_{n-1} + F_{n-2}$, $n > 2$ with $F_1 = F_2 = 1$ (which is equivalent to (1.5)). It is clear that the series on the right of (1.6) is obtained by expanding the rational function in the middle. Conversely, we can recover the rational function by manipulating the defining recurrence for F_n.

Multiply the recurrence by x^n and sum over $n > 2$ to obtain

$$\begin{aligned}\sum_{n=3}^{\infty} F_n x^n &= \sum_{n=3}^{\infty} F_{n-1} x^n + \sum_{n=3}^{\infty} F_{n-2} x^n \\ &= x \sum_{n=3}^{\infty} F_{n-1} x^{n-1} + x^2 \sum_{n=3}^{\infty} F_{n-2} x^{n-2} \\ &= x \sum_{n=2}^{\infty} F_n x^n + x^2 \sum_{n=1}^{\infty} F_n x^n\end{aligned}$$

So, if we let $f(x) = \sum_{n=1}^{\infty} F_n x^n$, and apply the initial values $F_1 = F_2 = 1$, we obtain

$$f(x) - F_1 x - F_2 x^2 = x(f(x) - F_1 x) + x^2 f(x)$$

or

$$f(x) - x - x^2 = x f(x) - x^2 + x^2 f(x)$$

or

$$(1 - x - x^2) f(x) = x + x^2 - x^2$$

Hence

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

There are numerous applications of binary strings. However, in this project we only consider applications of binary strings to partitions and compositions of numbers and we specifically study the results obtained. The analysis of the results will lead us to some known combinatorial identities with certain Fibonacci types.

1.1.1 Literature Review

The study of binary sequences has an extensive history. Thomas Harriot, Juan Caramuel y Lobkowitz and Gottfried Leibniz studied the modern binary number system in the sixteenth and seventeenth centuries. However, the numeration system related to binary numbers came into sight earlier in diverse cultures and societies including ancient Egypt, China and India. Leibniz was especially inspired by the Chinese I Ching [15], an ancient Chinese book of divination and a source of Confucian and Taoist philosophies which dates back to the ninth century BC. The binary notation in the I Ching is utilized to elucidate its Quaternary Divination. It depends on Taoistic duality of yin and yang. Eight trigrams (Bagua) and a set of 64 hexagrams ("64" gua), undifferentiated from the three-bit and six-bit binary numerals, were being used in some measure as early as the Zhou Dynasty of old China.

The Song Dynasty researcher Shao Yong (1011-1077) improved the hexagrams in a format that looks like current binary numbers, despite the fact that he didn't expect his plan to be utilized mathematically (see [30]). Viewing the most un-critical bit on top of single hexagrams in Shao Yong's square and examining along columns either from base right to upper left with strong lines as 0 and broken lines as 1 or from upper left to base right with strong lines as 1 and broken lines as 0 hexagrams can be deciphered as grouping from 0 to 63 [19].

Figure 1.1: A diagram showing Bagua arrangement

The scribes of ancient Egypt utilized something known as the Horus-Eye fractions (alleged in light of the fact that numerous historians of mathematics believed that the images utilized for this system could be set up to shape the eye of Horus, although this has been disputed) [32] which was one of the two distinct systems to represent their fractions. Horus-Eye fractions are a binary numbering system for fractional amounts of grain, fluids, or different measures, in which a small portion of a hekat(an ancient Egyptian volume unit used to measure) is conveyed as an amount of the paired fractions $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{32}, \frac{1}{64}$. Early types of this system can be found in archives from the Fifth Dynasty of Egypt, roughly 2400 BC, and its completely evolved hieroglyphic structure dates to the Nineteenth Dynasty of Egypt, around 1200 BC [7].

The technique utilized for antiquated Egyptian increase is likewise firmly connected with binary numbers. In this method, increasing one number by a second is performed by an arrangement of steps in which a value is either multiplied or has the initial number added once more into it; the order wherein these steps are to be carried out is given by the binary representation of the subsequent number. The method is used in the Rhind Mathematical Papyrus which dates to around 1650BC (see [33]).

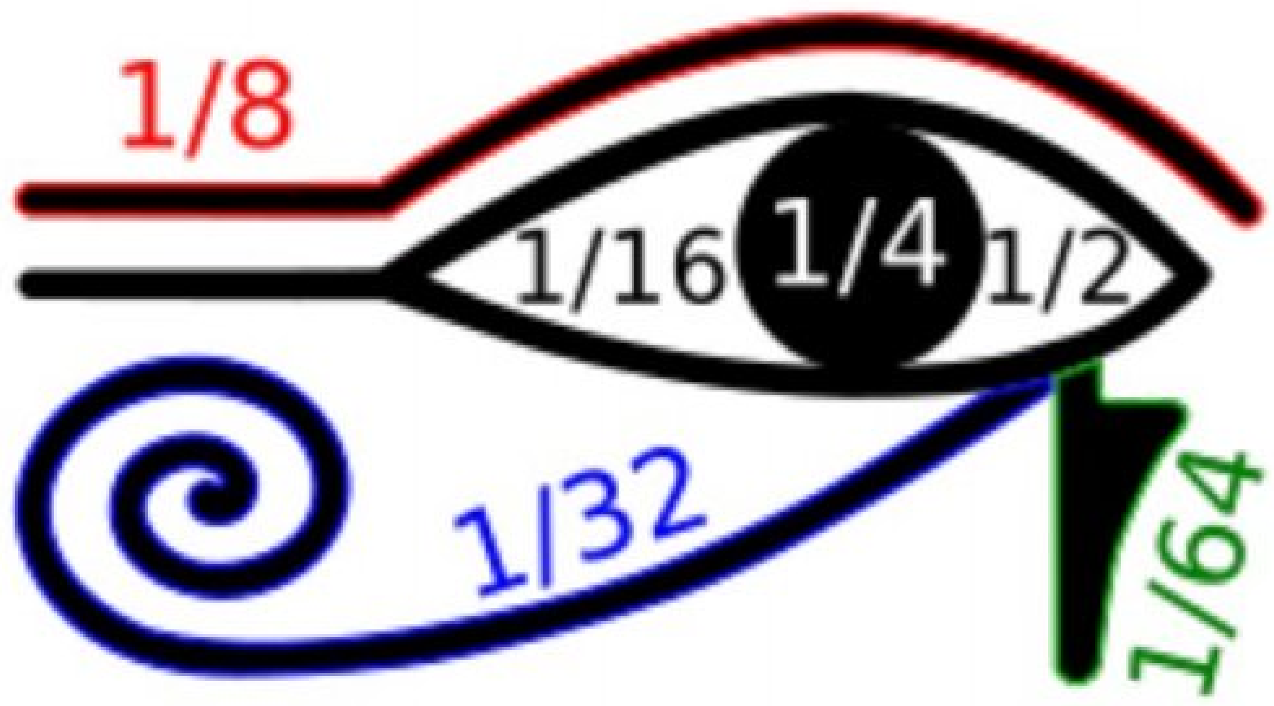

Figure 1.2: An illustration of the Horus-Eye with values

The Indian researcher Pingala developed a binary system around the 3rd and 2nd centuries BCE for portraying prosody [3], which is defined as the study of all the elements of language that contribute toward acoustic and rhythmic effects. He utilized binary numbers as short and long syllables (the last option equivalent long to two short syllables) [41], relating it to Morse code which is defined as either of two systems for representing letters of the alphabet, numerals, and punctuation marks by an arrangement of dots, dashes and spaces according to Britannica [5]. They were known as laghu (light) and guru (heavy) syllables.

Pingala's Hindu classic named Chandahśāstra depicts the development of a network to give a novel value to each meter. Chandahśāstra in a real sense means study of meters in Sanskrit. The binary portrayals in Pingala's system increments towards the right, and not to the left like in the binary numbers of the cutting edge positional notation[37]. In Pingala's system, the numbers start from number one, and not zero. Four short syllables 0000 is the primary example and relates to the value one. The numerical value is acquired by adding one to the sum of place values [41].

1.1.2 Importance of Binary sequences in Computer Science and Cryptography

Binary sequences are significant in our everyday lives especially in computer science. In the history of technology, binary number system is one of the most important developments. Binary sequences are significant in the light of the fact that utilizing them rather than the decimal system improves the efficiency of computer operations and similar innovations. They are also relatively less expensive. It is easier to store or manipulate binaries since the only possibilities for each digit are zero and one rather than the ten digits used in the decimal system. Claude Shannon [38] introduced the term *bit*, defined as the major unit of memory inside a computer. We use bits to assign any value that we can imagine. For instance, images can be represented by a two-dimensional array of bits and videos are formed from sequences of images. Sequences of bits have no essential meaning other than the description that we assign to them.

Cryptography alludes to the science and art of changing messages to make them secure and not susceptible to assaults. Essentially, it depends on the idea of abstract algebra. Network security is more accomplished through the use of cryptography. The cryptographic algorithms are mathematically dependent and work with numbers. Hence, we need to change a text to numbers before using algorithms and then change numbers back to text. For example, in One-Time Pad cryptography [1] (or [43]), a 5-bit binary method was utilized to change strings to binary sequences. For example, letters of the English alphabet can be changed into binary sequences as follows:

$$a \to 0 \to 00000\, b \to 01 \to 000001, ...z \to 25 \longrightarrow 11001.$$

Chapter 2

Enumeration of Binary Strings I: The Symbolic Method

In this chapter we define and study the theorems used for enumerating binary string using the symbolic method. We further give illustrations of the method using restricted examples.

2.1 Basic Symbolic Method

We give the essential definitions and notations.

Definition 2.1.1. A **block** in binary string is defined as a maximal nonempty substring of consecutive zeros and/or ones.

Definition 2.1.2. A **block decomposition** of a string is defined as partitioning of every string into its blocks in a unique way.

Definition 2.1.3. Concatenation of binary strings a and b is the binary string ab consisting of the entries of a followed by the entries of b. If A and B are sets of binary strings, then

$$AB := \{ab : a \in A, b \in B\} \tag{2.1}$$

is a concatenation of A and B.

Definition 2.1.4. The set of all concatenations of strings in A together with the empty string (ϵ) is defined by,

$$A^* := \{\epsilon\} \cup A \cup AA \cup AAA \cup \cdots$$

2.1.1 Ambiguity in Symbolic Method

The set $\{0,1\}^*$ is the set of all binary strings. Also note that $(\{0\}^*\{1\}^*)^*$ is the set of all binary strings. The strings in the set $\{0\}^*\{1\}^*$ are

$$\epsilon, 0, 1, 00, 11, 01, 000, 001, 011, 111, \ldots$$

We say that a string involving concatenation and the * operation is uniquely created if it can be created in only one way. For example, the string 101 can be created in two ways from AB when $A = \{10, 1\}$ and $B = \{1, 01\}$: $10 \in A$ can be concatenated with $1 \in B$ or $1 \in A$ with $01 \in B$. On the other hand, 101 is created in one way from AB when $A = \{10, 1\}$ and $B = \{0, 01\}$.
There are many ways to create a string from $(\{0\}^*\{1\}^*)^*$; for example the string 011 can be created as concatenation of 01 and 1 or 0 and 11. Evidently, $\{0,1\}^*$ uniquely creates all binary strings but $(\{0\}^*\{1\}^*)^*$ does not.

Now, we give a theorem in which blocks are restricted to ones only or zeros only.

Theorem 2.1.1. *The strings in the identity below*

$$\{0,1\}^* = \{1\}^*(\{0\}\{1\}^*)^* \qquad \textit{(used to condition on blocks of zeros only)} \qquad (2.2)$$

$$= \{0\}^*(\{1\}\{0\}^*)^* \qquad \textit{(used to condition on blocks of ones only)} \qquad (2.3)$$

are created uniquely.

Note that by conditioning on a string of blocks of 0's we mean that the string is decomposed after each occurrence of 0, and by conditioning on blocks of 1's we mean that the string is decomposed after each occurrence of 1.

Proof of Theorem 2.1.1. Consider $\{1\}^*(\{0\}\{1\}^*)^*$, the second is similar. Every string $a \in \{0,1\}^*$ can be represented uniquely as $a_1 0 a_2 0 a_3 \cdots 0 a_k$ where a_i is a block of ones or $a_i = \epsilon$. For any $i \in \{1\}^*$, we see that $0a_i \in \{0\}\{1\}^*$ and so $a \in \{1\}^*(\{0\}\{1\}^*)^*$ is uniquely created. □

Let S be a set of binary strings, and define $\omega(\sigma) = \ell(\sigma)$, as the length of σ for $\sigma \in S$. The symbol $\phi_s(x)$ denotes the generating function for this set of strings.

The following theorem may be used to find the generating function when two or more sets are concatenated.

Theorem 2.1.2. *Let A and B be sets of binary strings. If the elements of AB are uniquely created, then*

$$\phi_{AB}(x) = \phi_A(x) \cdot \phi_B(x) \ . \qquad (2.4)$$

If the elements of A^* are uniquely created, then

$$\phi_{A^*}(x) = \frac{1}{1-\phi_A(x)} \ . \tag{2.5}$$

Proof. Let $A^k = AAA\cdots A$ (k times), where $A^0 = \{\epsilon\}$. Since $A^* = \{\epsilon\} \cup A \cup AA \cup \cdots$. If we can show that Equation (2.4) holds then Equation (2.5) follows because

$$\phi_{A^*}(x) = \sum_{k=0}^{\infty} \phi_{A^k}(x) = \sum_{k=0}^{\infty} \phi_A(x)^k \ . \tag{2.6}$$

This gives the geometric series, which gives the second identity. For the first identity,

$$\phi_{AB}(x) = \sum_{ab\in AB} x^{\omega(ab)} = \sum_{a\in A}\sum_{b\in B} x^{\omega(a)}x^{\omega(b)} = \phi_A(x)\phi_B(x)$$

since $\omega(ab) = \omega(a) + \omega(b)$. □

Theorem 2.1.3. *The strings in the identities below are uniquely created*

$$\{0,1\}^* = \{0\}^*(\{1\}\{1\}^*\{0\}\{0\}^*)^*\{1\}^* \tag{2.7}$$
$$= \{1\}^*(\{0\}\{0\}^*\{1\}\{1\}^*)^*\{0\}^* \tag{2.8}$$

We note that equation (2.2) is used to condition on restrictions of blocks of ones only and equation (2.3) is used to condition on restrictions of blocks of zeros only. Whereas equation (2.7) and (2.8) are used to condition on restrictions of blocks of ones and zeros simultaneously.

We also notice that equation (2.7) (or 2.8 respectively) have the middle part $\{1\}\{1\}^*\{0\}\{0\}^*$ (or $\{0\}\{0\}^*\{1\}\{1\}^*$) and end-parts $\{0\}^*$ and $\{1\}^*$ (or $\{1\}^*$ and $\{0\}^*$). We generally construct the middle part $M(01)$, and a pair of right and left end-parts $L(01)$ and $R(01)$, to give

$$L(01)(M(01))^*R(01).$$

Also, the middle part $M(01)$ can be constructed using its complement $M(01)'$ as follows;

$M(01) = \{0\}\{0\}^*\{1\}\{1\}^* \setminus M(01)'$ (by the middle part of equation (2.7) (or (2.8))

Note that $M(01)$ is expected to repeat an arbitrary number of times by construction, thus it is represented by $M(01)^*$.

We now give illustrations of the above theorems.

We use the set S to denote the set of binary strings in the following examples.

Example 2.1.1. Consider the set of binary strings that start and end with 1.

Decomposition of n-bit strings of the type $(1, ...1)$ is $\{1\}(\{0,1\})^*\{1\}$ (since the condition is restricted to blocks of 1's only).

We use Theorem 2.1.2 to find the generating function,

$$\begin{aligned}\phi_{S(x)} &= \phi_{\{1\}(\{0,1\}^*\{1\}}(x)\\ &= \phi_{\{1\}}(x)\phi_{\{0,1\}^*}(x)\phi_{\{1\}}(x)\\ &= \phi_{\{1\}}(x)\frac{1}{1-\phi_{\{0,1\}}}\phi_{\{1\}}\\ &= x\frac{1}{1-2x}x\\ &= \frac{x^2}{1-2x}\ .\end{aligned}$$

Therefore the number of strings of length n that begin and end with 1 is

$$[x^n]\phi_{S(x)} = 2^{n-2}.$$

Example 2.1.2. Consider counting bit strings that avoid 0011.

We use Theorem 2.1.3 to construct the middle part $M(01)$ which generates strings with one 0 followed by any number of 1's or strings with two or more 0's followed by a single 1. That is,

$$M(01) = \{0\}\{1\}\{1\}^* \cup \{00\}\{0\}^*\{1\}.$$

Thus the full decomposition of such strings, after appending possible first and last bits, is

$$\{1\}^*(\{0\}\{1\}\{1\}^* \cup \{00\}\{0\}^*\{1\})^*\{0\}^*. \tag{2.9}$$

The contribution of the middle factor in (2.9) to the final generation function of $S(x)$ is

$$\left(1-\left(x\cdot x\cdot\frac{1}{1-x}+x^2\cdot\frac{1}{1-x}\cdot x\right)\right)^{-1}.$$

Note that the operation * indicates reciprocal.
Hence the generating function of $S(x)$ is given by,

$$\begin{aligned} S(x) &= \phi_{\{1\}^*(\{0\}\{1\}\{1\}^* \cup \{00\}\{0\}^*\{1\})^*\{0\}^*}(x) \\ &= \frac{1}{1-x} \cdot \frac{1}{1-(\frac{x^2}{1-x}+\frac{x^2}{1-x}x)} \cdot \frac{1}{1-x} \\ &= \frac{1}{1-x} \cdot \frac{1}{1-x-x^2-x^3} \\ &= \frac{1}{1-2x+x^4}. \end{aligned}$$

The recurrence relation for $c(n)$ may be obtained as follows:

$$\frac{1}{1-2x+x^4} = \sum_{n=0}^{\infty} c(n)x^n$$

which gives

$$\begin{aligned} 1 &= (1-2x+x^4)\sum_{n=0}^{\infty} c(n)x^n \\ &= (1-2x+x^4)\sum_{n=0}^{\infty} c(n)x^n \\ &= \sum_{n=0}^{\infty} c(n)x^n - 2x\sum_{n=0}^{\infty} c(n)x^n + x^4\sum_{n=0}^{\infty} c(n)x^n \\ &= \sum_{n=0}^{\infty} c(n)x^n - 2\sum_{n=0}^{\infty} c(n)x^{n+1} + \sum_{n=0}^{\infty} c(n)x^{n+4}. \end{aligned}$$

Therefore,

$$\sum_{n=0}^{\infty} c(n)x^n = 1 + 2\sum_{n=1}^{\infty} c(n-1)x^n - \sum_{n=4}^{\infty} c(n-4)x^n.$$

Thus the number $c(n)$ of bit strings that avoid 0011 is given by the recurrence relation,

$$c(n) = 2c(n-1) - c(n-4),\ n \geq 4. \qquad (2.10)$$

with $c(0) = 1$, $c(1) = 2$, $c(2) = 4$, $c(3) = 8$.

Now, suppose we want to count bit strings that avoid 1100.

Such strings can be decomposed using Theorem 2.1.3 with the complement of the middle part $M(01)$. In this case the complement $M(01)'$ is the set of bit strings that contain 1100 which is given by

$$M(01)' = \{11\}\{1\}^*\{00\}\{0\}^*.$$

Therefore we have

$$\begin{aligned} M(01) &= \{0\}\{0\}^*\{1\}\{1\}^* \setminus M(01)' \\ &= \{0\}\{0\}^*\{1\}\{1\}^* \setminus \{11\}\{1\}^*\{00\}\{0\}^*. \end{aligned}$$

Thus the required full decomposition is

$$\{1\}^*(\{0\}\{0\}^*\{1\}\{1\}^* \setminus \{11\}\{1\}^*\{00\}\{0\}^*)^*\{0\}^*.$$

Hence, the generating function for the enumerated set $S(x)$ is

$$\begin{aligned} \phi_{S(x)} &= \phi_{\{1\}^*(\{0\}\{0\}^*\{1\}\{1\}^*\setminus\{11\}\{1\}^*\{00\}\{0\}^*)^*\{0\}^*} \\ &= \frac{1}{1-x}\frac{1}{1-(\frac{x^2}{(1-x)^2}-\frac{x^4}{(1-x)^2})}\frac{1}{1-x} \\ &= \frac{1}{1-2x+x^4} \end{aligned}$$

It is not surprising that the generating function for bits avoiding 1100 is equal to the generating function for bits avoiding 0011 as both strings are *complementary*, that is, one may be obtained from the other by interchanging 0's and 1's. Thus the number of bit strings avoiding 1100 is also given by $c(n)$, see (2.10).

Example 2.1.3. Suppose we want to find the number of bit strings that avoid 100.

We use Theorem 2.1.3 and construct the middle part $M(01)$ using its complement. Since the set of bit strings that contain 100 is given by $\{1\}\{1\}^*\{00\}\{0\}^*$ we obtain

$$M(01) = \{0\}\{0\}^*\{1\}\{1\}^* \setminus \{1\}\{1\}^*\{00\}\{0\}^*.$$

Thus the decomposition of such strings becomes,

$$\{1\}^*(\{0\}\{0\}^*\{1\}\{1\}^* \setminus \{1\}\{1\}^*\{00\}\{0\}^*)^*\{0\}^*.$$

Hence the generating function of $S(x)$ is given by,

$$\begin{aligned}\phi_{S(x)} &= \phi_{\{1\}^*(\{0\}\{0\}^*\{1\}\{1\}^*\setminus\{1\}\{1\}^*\{00\}\{0\}^*)\{0\}^*}\\ &= \frac{1}{1-x}\frac{1}{1-(\frac{x^2}{(1-x)^2}-\frac{x^3}{(1-x)^2})}\frac{1}{1-x}\\ &= \frac{1}{1-2x+x^3}\end{aligned}$$

Thus the number of bit strings that avoid 100 is given by the linear recurrence relation,

$$c(n) = 2c(n-1) - c(n-3)$$

with $c(0) = 1$, $c(1) = 2$, $c(2) = 4$, $c(3) = 7$ and $n \geq 3$.

Note that

$$\begin{aligned}\sum_{n=0}^{\infty} c(n)x^n &= \frac{1}{1-2x+x^3} = \frac{1}{1-x}\cdot\frac{1}{1-x-x^2}\\ &= \left(\sum_{m=0}^{\infty} x^m\right)\left(\sum_{i=0}^{\infty} F_i x^i\right)\\ &= \sum_{n=0}^{\infty}\left(\sum_{m=0}^{n} F_{n-m}\right)x^n \qquad\qquad (m+i=n)\end{aligned}$$

Thus extracting coefficients of x^n yields in terms of the Fibonacci numbers minus 1:

$$c(n) = \sum_{m=0}^{n} F_{n-m} = F_n - 1, \quad n \geq 2.$$

Example 2.1.4. Consider counting bit strings without odd runs of zeros.

Such strings can be decomposed directly by applying Theorem 2.1.3 with even blocks of zeros. That is,

$$\{1\}^*(\{00\}\{00\}^*\{1\}\{1\}^*)^*\{00\}^*.$$

Hence the generating function of $S(x)$ is given by

$$\begin{aligned}\phi_{S(x)} &= \frac{1}{1-x} \cdot \frac{1}{1-\frac{x^2}{1-x^2}\cdot\frac{x}{1-x}} \cdot \frac{1}{1-x^2} \\ &= \frac{1}{(1-x)(1-x^2)} \cdot \frac{(1-x)(1-x^2)}{(1-x^2)(1-x)-x^3} \\ &= \frac{1}{1-x-x^2}\end{aligned}$$

Thus the number of bit strings avoiding odd runs of zeros is F_{n+1}.

Example 2.1.5. Consider counting bit strings which do not contain 101.

This appears to be a more tricky example by directly applying the symbolic method so we shall partition the set, S of strings avoiding 101 into three parts, namely S_1, S_0 and strings with no double blocks denoted by S_n. (Note: a double block means a non-empty string of 0's followed by a non-empty string of 1's).

Firstly, let S_1 be strings which starts with 1 and has a double block. Then S_1 can be decomposed as follows,

$$\{1\}\{1\}^*(\{00\}\{0\}^*\{1\}\{1\}^*)(\{00\}\{0\}^*\{1\}\{1\}^*)^*\{0\}^*$$

since every double block in S_1 starts with at least two zeros.

Secondly, let S_0 be strings which start with a zero. Then S_0 can be decomposed as follows,

$$\{\epsilon\}(\{0\}\{0\}^*\{1\}\{1\}^*)(\{00\}\{0\}^*\{1\}\{1\}^*)^*\{0\}^*$$

since the first two blocks are arbitrary and the ones after that have one zero.

Lastly the strings with no double blocks cannot contain 101 since their decomposition is $\{1\}^*\{\epsilon\}\{0\}^*$.

Thus the decomposition of $S(x)$ is given by

$$S = S_1 \cup S_0 \cup S_n$$

Hence, the generating function of $S(x)$ is given by,

$$\begin{aligned}\phi_{S(x)} &= \frac{x^4}{(1-x)^4} \cdot \frac{1}{1-\frac{x^3}{(1-x)^2}} + \frac{x^2}{(1-x)^3} \cdot \frac{1}{1-\frac{x^3}{(1-x)^2}} + \frac{1}{(1-x)^2} \\ &= \frac{x^4}{(1-x)^2(1-2x+x^2-x^3)} + \frac{x^2}{(1-x)(1-2x+x^2-x^3)} + \frac{1}{(1-x)^2} \\ &= \frac{1-2x+2x^2-2x^3+x^4}{(1-x)^2(1-2x+x^2-x^3)} \\ &= \frac{x^2+1}{1-2x+x^2-x^3}\end{aligned}$$

Thus the number of bit strings avoiding 101 is given by recurrence relation,

$$a_n = 2a_{n-1} - a_{n-2} + a_{n-3}, \qquad \text{for } n \geq 3 \tag{2.11}$$

with $a_0 = 1$, $a_1 = 2$ and $a_2 = 4$.

Theorem 2.1.4. *The number of bit strings of length n avoiding 010 is equal to the number of bit strings of length $(n+1)$ avoiding isolated 1's.*

Proof. We use generating functions to show that the number of bit strings of length n avoiding 101 is equal to the number of bit strings of length $(n+1)$ avoiding isolated 1's. Now, we begin with the bit string of length n avoiding isolated 1's. The decomposition of bit string of length n avoiding isolated ones is

$$\{\epsilon, 11, 111, ...\}(\{0\}\{\epsilon, 11, 111, ...\}^*)^*$$

Thus the generating function for such strings is given by

$$\begin{aligned}\phi(x) &= \frac{1 + x^2 + x^3 + ...}{1 - x(1 + x^2 + x^3 + ...} \\ &= \frac{1 + \frac{x^2}{1-x}}{1 - x(1 + \frac{x^2}{1-x}} \\ &= \frac{1 - x + x^2}{1 - x - x + x^2 - x^3} \\ &= \frac{1 - x + x^2}{1 - 2x + x^2 - x^3}.\end{aligned}$$

Recall that in Section 2, example 2.1.5 we showed that the total number of bit strings of length n avoiding 101 is given by the generating function $\frac{1+x^2}{1-2x+x^2-x^3}$. Note that

$$A(x) := \frac{1 - x + x^2}{1 - 2x + x^2 - x^3} = 1 + x + 2x^2 + 4x^3 + 7x^4 + 12x^5 + 21x^6 + 37x^7 + \cdots$$

while

$$E(x) := \frac{1 + x^2}{1 - 2x + x^2 - x^3} = 1 + 2x + 4x^2 + 7x^3 + 12x^4 + 21x^5 + 37x^6 + 65x^7 + \cdots .$$

Then it can be verified that $A(x) = 1 + xB(x)$.

Denoting the coefficient of x^n in these two generating functions by $a(n)$ and $b(n)$ respectively, we see that $a(n+1) = b(n)$. Thus the number of bit strings avoiding

101 and the number of bit strings avoiding isolated 1's have essentially the same generating function.

Thus the two enumeration functions also satisfy the same recurrence relations, subject to a shift in indices. (See an example below). □

Example 2.1.6. Let $n = 4$. Then $a(5) = 12 = b(4)$. The bit strings of length 5 avoiding isolated 1's which are counted by $a(5) = 12$, are

$$1^5,\ 01^4,\ 0^21^3,\ 1^201^2,\ 0^31^2,\ 1^40,\ 01^30,\ 1^21^20,\ 1^30^2,\ 01^20^2,\ 1^20^3,\ 0^5.$$

The bit strings of length 4 avoiding 101 which are counted by $b(4) = 12$, are

$$0^4,\ 10^3,\ 010^2,\ 1^20^2,\ 0^210,\ 01^20,\ 1^30,\ 0^31,\ 10^21,\ 0^21^2,\ 01^3,\ 1^4.$$

Chapter 3

Enumeration of Binary strings II: Other Methods

In this section we focus on other methods of enumerating binary strings namely, primitive recursion and direct constructions. We give examples with restricted bit strings.

3.1 Primitive Recursion

A recurrence relation is an equation that recursively defines sequences.

Definition 3.1.1. Given a sequence $\{a_n\}_{n=o}^{\infty}$ we define the recurrence relation for $\{a_n\}_{n=o}^{\infty}$ as an equation which associates the nth term a_n to some of its previous terms.

Examples of recurrence relations

Example 3.1.1. Consider counting binary strings of length $n \geq 0$ that avoids consecutive 0's.

Such binary strings can be counted as follows;

Let the set of binary strings that do not contain consecutive zeros be denoted by S_n and $|S_n| = a_n$ denote the cardinality of S_n.

For example,

$S_0 = \{\epsilon\}$ thus $|S_0| = a_0 = 1$ (empty set)
$S_1 = \{0, 1\}$ thus $|S_1| = a_1 = 2$
$S_2 = \{01, 10, 11\}$ thus $|S_2| = a_2 = 3$
$S_3 = \{010, 110, 101, 011, 111\}$ thus $|S_3| = a_3 = 5$
$S_4 = \{1010, 0110, 1110, 0101, 1101, 1011, 0111, 1111\}$, thus $|S_4| = a_4 = 8$

We note that from the first five values of a_n we suspect the Fibonacci numbers. Thus we claim that the numbers a_n satisfy the recurrence relation.

$$a_n = a_{n-1} + a_{n-2}, \qquad n \geq 2 \tag{3.1}$$

with $a_1 = 1$ and $a_2 = 2$

Proof. Let the string counted by a_n be denoted by A. That is, $A \in S_n$. We build A by first positioning 1 at the start of any $T \in S_{n-1}$ to obtain A in a_{n-1} ways. Secondly, when A starts with 0 it can be built by positioning 01 at the starting point of a string of length $n - 2$ to obtain A in a_{n-2} ways.

Hence A can be built in as many ways as

$$a_{n-1} + a_{n-2},$$

which yields the main results with initial conditions $a_0 = 1$, $a_1 = 2$.

The generating function can be found as follows. Let

$$T(x) = \sum_{n=0}^{\infty} a_n x^n.$$

Multiplying through (3.1) by x^n and summing over $n \geq 2$ gives

$$\sum_{n=2}^{\infty} a_n x^n = \sum_{n=2}^{\infty} a_{n-1} x^n + \sum_{n=2}^{\infty} a_{n-2} x^n.$$

Shift summation indices to get,

$$\sum_{n=0}^{\infty} a_n x^n - 1x^0 - 2x = x\left(\sum_{n=0}^{\infty} a_n x^n - 1x^0\right) + x^2 \sum_{n=0}^{\infty} a_n x^n.$$

Therefore,

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

This implies

$$T(x) = \frac{1+x}{1-x-x^2} \; . \tag{3.2}$$

Now,

$$\begin{aligned} T(x) &= \frac{1}{1-x-x^2} + \frac{x}{1-x-x^2} \\ &= \sum_{n=0}^{\infty} F_{n+1}x^n + \sum_{n=1}^{\infty} F_n x^n \\ &= \sum_{n=0}^{\infty} (F_{n+1} + F_n)x^n \\ &= \sum_{n=0}^{\infty} F_{n+2}x^n \; . \end{aligned}$$

Therefore

$$a_n = [x_n]T(x) = F_{n+2}.$$

Thus the number of strings of length $n \geq 0$ that avoids consecutive zeros is F_{n+2}. □

Example 3.1.2. Suppose we wish to count the number of bit strings of length n that contain at least one pair of consecutive zeros.

Let A_n be the set of strings of length n that contain a pair of consecutive zeros and let $|A_n| = a_n$. Now, let A be an object enumerated by a_n. Then A can be constructed in three ways. If we insert 1 at the beginning of any $B \in A_{n-1}$ then the bit string of length $n-1$ has a_{n-1} possible strings that contain a pair of consecutive zeros. If A starts with 0, it can be built by affixing a 01 at the beginning of a string of length $n-2$ to obtain A in a_{n-2} ways. Lastly, if A starts with 00 then there are 2^{n-2} such strings of length $n-2$ followed by 00. Thus adding the cases yields a linear recurrence,

$$a_n = a_{n-1} + a_{n-2} + 2^{n-2}$$

When $n = 0$, $a_0 = 0$.
When $n = 1$, $a_1 = 0$.

When $n = 2$, $a_2 = 1$ thus the recurrence holds for $n = 2$.
Therefore,

$$a_n = a_{n-1} + a_{n-2} + 2^{n-2} \qquad n \geq 2, \tag{3.3}$$

with $a_0 = 0 = a_1$.

The generating function of bit strings containing a pair of consecutive zeros can be found as follows.

Let

$$T(x) = \sum_{n=0}^{\infty} a_n x^n.$$

Multiplying through (3.3) by x^n and summing over $n \geq 2$,

$$\sum_{n=2}^{\infty} a_n x^n = \sum_{n=2}^{\infty} a_{n-1} x^n + \sum_{n=2}^{\infty} a_{n-2} x^n + \sum_{n=2}^{\infty} 2^{n-2} x^n.$$

Shift summation indices to obtain,

$$\sum_{n=0}^{\infty} a_n x^n = \sum_{n=0}^{\infty} a_n x^{n+1} + \sum_{n=0}^{\infty} a_n x^{n+2} + \sum_{n=0}^{\infty} 2^n x^{n+2}.$$

Introducing $T(x)$ yields,

$$T(x) = x \cdot T(x) + x^2 \cdot T(x) + \frac{x^2}{1-2x}.$$

This implies,

$$\begin{aligned} T(x)(1 - x - x^2) &= \frac{x^2}{1-2x} \\ T(x) &= \frac{x^2}{(1-2x)(1-x-x^2)} \end{aligned}$$

Using partial fractions for $T(x)$ yields

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