Regular Constructions

Contents

1 Recall

Definition. The ordinary generating function (**OGF**) of a sequence (A_n) is the formal power series

$$A(z) = \sum_{n=0}^{\infty} A_n z^n.$$

We extend this to say that the OGF of a class $\mathcal{A}$ is the generating function of its counting sequence $(A_n)_{n\geq 0}$, where $A_n = \text{cardinality}(\mathcal{A}_n)$. Equivalently, we can write the OGF as

$$A(z) = \sum_{\alpha\in\mathcal{A}} z^{|\alpha|}.$$

In this context we say that the variable z **marks** the size of the underlying objects.

Two formal power series facts that we shall use: Multiplication rule:

$$\left(\sum A_n z^n\right)\left(\sum B_n z^n\right) = \sum_n \left(\sum_{k=0}^{n} A_k B_{n-k}\right) z^n$$

Quasi-inverse: If $A(z)$ is such that $A_0 = 0$, then

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

2 A calculus for combinatorial classes

2.1 The basic idea: Union ≡ sum

Our approach builds up a new class from existing classes, and translates counting information at the same time. A simple example of this principle is as follows. Imagine class $\mathcal{A}$ and class $\mathcal{B}$ have empty intersection, and that we know everything about them. What can we say about $\mathcal{C}$, if it is the union of these two classes:

$$\mathcal{C} = \mathcal{A} \cup \mathcal{B}?$$

In particular, $\mathcal{C}_n = \mathcal{A}_n \cup \mathcal{B}_n$, and hence $C_n = A_n + B_n$. This only works if the union is **disjoint**, i.e. $\mathcal{A} \cap \mathcal{B} = \emptyset$, but when this does hold, no further information is needed about $\mathcal{A}$ or $\mathcal{B}$. We translate the structural information about $\mathcal{C}$ into counting information about $\mathcal{C}$. Similarly, to describe a random generation scheme for $\mathcal{C}$, it is sufficient to have one for $\mathcal{A}$ and $\mathcal{B}$, and a way to choose which one to have an element from. But we are getting ahead of ourselves.

2.2 Admissible constructions

Now we build a toolbox of standard constructions. Informally, a construction is a rule Φ which inputs a sequence $\mathcal{B}^{(1)}, \ldots, \mathcal{B}^{(m)}$ of combinatorial classes, and produces a new combinatorial class

$$\mathcal{A} = \Phi\left(\mathcal{B}^{(1)}, \ldots, \mathcal{B}^{(m)}\right).$$

We want to be able to find the counting sequence for $\mathcal{A}$ by just knowing the counting sequences for $\mathcal{B}^{(i)}, i = 1, 2, ..., m$. We will define a small set of constructions from which we can derive a remarkable number of objects of interest.

First, we give a technical definition which formalizes the above intuition.

Definition. Let Φ be an m-ary construction that associates to any sequence of classes $\mathcal{B}^{(1)}, \ldots, \mathcal{B}^{(m)}$ a new combinatorial class $\mathcal{A} = \Phi\left(\mathcal{B}^{(1)}, \ldots, \mathcal{B}^{(m)}\right)$. The construction Φ is **admissible** if the counting sequence (A_n) only depends on the counting sequences $\left(B_n^{(1)}\right), \ldots, \left(B_n^{(m)}\right)$.

Our toolbox of combinatorial constructions will include the following kinds of operations:

- union;
- cartesion product;
- sequence;
- set and multiset;
- cycle;
- pointing and substitution.

These "admissable constructions" will translate directly to simple (and not so simple) operations on the generating functions.

Example. Let $\mathcal{B}^{(1)}$, $\mathcal{B}^{(2)}$, $\mathcal{B}^{(3)}$ be three combinatorial classes. An example of a m-ary construction (with $m = 3$) that uses **union**, **cartesian product** and **sequence** (these will be defined below) is the new combinatorial class

$$\mathcal{A} = \Phi\left(\mathcal{B}^{(1)}, \mathcal{B}^{(2)}, \mathcal{B}^{(3)}\right) = \textsc{Seq}(\mathcal{B}^{(1)} \times (\mathcal{B}^{(2)} \cup (\mathcal{B}^{(3)} \times \mathcal{B}^{(1)}))$$

Given an admissible construction Φ, there exists a well-defined operator Ψ acting on the corresponding ordinary generating functions: $A(z) = \Psi\left(B^{(1)}(z), ..., B^{(m)}(z)\right)$.

Our constructions will be mechanical, but they will have a natural bijection with the desired class.

2.2.1 Bijective combinatorics

Enumeration is also important because it may help us recognize two classes that are essentially the same.

Definition. Two classes $\mathcal{A}, \mathcal{B}$ are said to be **combinatorially isomorphic**, written $\mathcal{A} \cong \mathcal{B}$ iff their counting sequences are the same. Note that $\mathcal{A} \cong \mathcal{B}$ iff there is a size-preserving bijection between $\mathcal{A}$ and $\mathcal{B}$.

There are many beautiful results in combinatorics resulting from the construction of bijections between seemingly disparate classes of objects. This "bijective" school of combinatorics is championed by the French.

We will describe classes that will be in very natural bijections with the objects we are truly interested in.

2.3 Cartesian Product

The easiest of these is the construction based on the cartesian product

Definition. The **cartesian product** construction applied to two classes $\mathcal{B}, \mathcal{C}$ results in a call $\mathcal{A}$ of ordered pairs,

$$\mathcal{A} = \mathcal{B} \times \mathcal{C} = \{\alpha = (\beta, \gamma) \mid \beta \in \mathcal{B}, \gamma \in \mathcal{C}\}$$

where the size of a pair is additive, namely

$$|\alpha|_{\mathcal{A}} = |\beta|_{\mathcal{B}} + |\gamma|_{\mathcal{C}}.$$

By considering all possible pairs we see that the counting sequences satisfy

$$A_n = \sum_{k=0}^{n} B_k C_{n-k}$$

which is exactly the product of the generating functions

$$A(z) = B(z) \cdot C(z).$$

Since the counting sequence (A_n) depends only the sequences (B_n) and (C_n) (and not any other information) we see that the cartesian product is admissible and it translates as a product of OGFs.

Exercise. Verify that the set $\mathcal{A}$ defined as $\mathcal{A} = \mathcal{B} \times \mathcal{C}$ is a well-defined as a combinatorial class. Understand why cartesian product is an admissible operator.

Example. Consider the class of binary strings with a break, or bookmark somewhere in the string, possibly at the beginning or the end. Length is still the number of 1s plus the number of 0s. For example $100101^{\vee}100100$. We can model this by a cartesian product of binary strings: $100101^{\vee}100100 \equiv (100101, 100100)$. Here the length is additive. Thus, this class is equivalent to $\mathcal{V} \equiv \mathcal{W} \times \mathcal{W}$. We count the number of elements of length n in three different ways:

1. Combinatorial argument First, we can see that for any usual binary string of length n, there are $n+1$ ways to insert the bookmark, including the beginning and the end, hence we expect $2^n(n+1)$ elements of length n.

2. Recurrence on coefficients From the above formula, we have

$$\mathcal{V} \equiv \mathcal{W} \times \mathcal{W} \implies V_n = \sum_{k=0}^{n} W_k W_{n-k} = \sum_{k=0}^{n} 2^k 2^{n-k} = \sum_{k=0}^{n} 2^n = (n+1)2^n,$$

as predicted.

3. Generating functions Using the fact $W(z) = \dfrac{1}{1-2z}$ and the identity $[z^n]\dfrac{1}{(1-mz)^r} = \dbinom{n+r-1}{r-1} m^r$, we have that

$$V_n = [z^n]W(z)^2 = [z^n]\frac{1}{(1-2z)^2} = \binom{2+n-1}{2-1} 2^n = (n+1)2^n.$$

Exercise. Which one of these methods is easiest if we have 2 bookmarks (ordered)? 200 bookmarks? Which one is most automatic? What if each word has either 2 or 3 bookmarks?

2.4 Revisiting Unions

Similarly we have seen that the **disjoint union** is nice

Definition. Let $\mathcal{A}, \mathcal{B}, \mathcal{C}$ be combinatorial classes such that

$$\mathcal{A} = \mathcal{B} \cup C \qquad\qquad \mathcal{B} \cap \mathcal{C} = \varnothing$$

with size defined in a compatible way; if $\omega \in A$ then

$$|\omega|_{\mathcal{A}} = \begin{cases} |\omega|_{\mathcal{B}} & \text{if } \omega \in \mathcal{B} \\ |\omega|_{\mathcal{C}} & \text{if } \omega \in \mathcal{C}. \end{cases}.$$

In this case, we may use the "sum notation" $\mathcal{A} = \mathcal{B} + \mathcal{C}$ instead of $\mathcal{A} = \mathcal{B} \cup \mathcal{C}$. Then we clearly have

$$A_n = B_n + C_n$$
$$A(z) = B(z) + C(z)$$

Hence the union of disjoint sets is admissible and it translates as a sum of OGF.

If our problem is to count the elements of a union of disjoint sets or a cartesian product then we do not have to write down explicit recurrence relations as an intermediate stages, rather we can directly write down the relationship among the generating functions. This is the spirit of the **symbolic method for combinatorial enumeration**.

2.5 Elemental classes

So let us prove a theorem demonstrating some of the basic constructions we can use to build up more complicated objects.

Definition (Neutral + atomic classes)**.** Let $\mathcal{E}$ denote the **neutral class** that consists of a single object of size zero. This is typically denoted ϵ. It typically denotes (say) the single empty word, the single graph of zero vertices, or single graph of zero edges etc. In particular, the neutral class satisfies

$$\mathcal{E} = \{\epsilon\} \quad \text{and} \quad E(z) = 1.$$

Let $\mathcal{Z}$ denote the **atomic class** that consists of a single object of size 1 called an atom. This is typically something like a single generic vertex or a single generic edge or a single generic letter. Distinct copies of the atomic class should be denoted with a subscript. For example, for binary words we could have $\mathcal{Z}_0 = \{0\}$, $\mathcal{Z}_1 = \{1\}$ etc. Each atomic class $\mathcal{Z}$ satisfies

$$Z(z) = z.$$

A key way in which we use ϵ: elements of $\mathcal{E} \times \mathcal{B}$ are of the form (ϵ, β), and the size of this element is $|(\epsilon, \beta)| = |\epsilon| + |\beta| = |\beta|$. Thus we have a natural combinatorial isomorphism

$$\mathcal{B} \cong \mathcal{E} \times \mathcal{B}.$$

Cartesian product and disjoint union we have already seen. It is possible to to handle the "disjointedness" without explicitly knowing that the sets are disjoint. We use "+" instead of "∪". The idea is to do something like

$$\mathcal{B} + \mathcal{C} = (\{\text{blue}\} \times \mathcal{B}) \cup (\{\text{red}\} \times \mathcal{C})$$

That is, we colour all the elements of one set blue and the other set red — so that the size is unchanged and so that these new coloured sets must be disjoint regardless of whether or not the original sets are.

Note that if we really have to worry about whether or not the sets are disjoint we get into trouble because we need to compute the size of the intersection

$$\text{card}(\mathcal{B}_n \cup \mathcal{C}_n) = \text{card}(\mathcal{B}_n) + \text{card}(\mathcal{C}_n) - \text{card}(\mathcal{B}_n \cap \mathcal{C}_n)$$

where $\text{card}(\mathcal{B}_n)$ is the cardinality of the set $\mathcal{B}_n$. We cannot derive the size of the intersection from the OGFs $B(z)$, $C(z)$ without additional information, so this is not an admissible construction.

Now, with just union and cartesian product we could define a few finite classes, but let us add one more to our toolbox before we move on to more advanced topics.

2.6 Sequence construction

Definition (Sequence)**.** If $\mathcal{A}$ is a class then the sequence class, then $\textsc{Seq}(\mathcal{A})$ is defined to be the infinite union

$$\textsc{Seq}(\mathcal{A}) = \mathcal{E} + \mathcal{A} + (\mathcal{A} \times \mathcal{A}) + (\mathcal{A} \times \mathcal{A} \times \mathcal{A}) + \dots .$$

Equivalently

$$\textsc{Seq}(\mathcal{A}) = \{(\alpha_1, \alpha_2, \dots, \alpha_\ell) \mid \ell \geq 0, \text{ and } \alpha_j \in \mathcal{A} \text{ for } j = 1, 2, \dots, \ell\}.$$

The size of an object $\beta \in \textsc{Seq}(\mathcal{A})$ is computed as follows.

$$\beta = (\alpha_1, \alpha_2, \dots, \alpha_\ell) \quad \Longrightarrow \quad |\beta| = |\alpha_1| + \cdots + |\alpha_\ell|.$$

It is common to use the "star notation", $A^* = \textsc{Seq}(\mathcal{A})$.

For example, the class of all binary strings is $\textsc{Seq}(\mathcal{Z}_0 + \mathcal{Z}_1) = \textsc{Seq}(\{0,1\}) = \{0,1\}^*$. Note that it is common the two atomic classes, $\mathcal{Z}_0$ and $\mathcal{Z}_1$, as 0 and 1

Danger! Take note that $\textsc{Seq}(\mathcal{A})$ is a combinatorial set only if $\mathcal{A}$ does not contain an element of size zero (a neutral element).

Exercise. Why does $\mathcal{A}^*$ fail to be a combinatorial set if $\epsilon \in A$?

Here are some variations, where the lengths of sequences from $\mathcal{A}$ have restricted. The following table gives some natural notations for these as well as their ogfs.

Description of class $\mathcal{B}$	Notation for $\mathcal{B}$	OGF $B(z)$
All sequences from $\mathcal{A}$	$\textsc{Seq}(\mathcal{A})$ or $\mathcal{A}^*$	$1 + A(z) + A(z)^2 + A(z)^3 + \cdots = \dfrac{1}{1 - A(z)}$
Sequences of length k from A	$\textsc{Seq}_{=k}(\mathcal{A})$ or $\mathcal{A}^k$	$A(z)^k$
Sequences of length at least k	$\textsc{Seq}_{\geq k}(\mathcal{A})$	$A(z)^k + A(z)^{k+1} + A(z)^{k+2} + \cdots = \dfrac{A(z)^k}{1 - A(z)}$
Sequences of length at most k	$\textsc{Seq}_{\leq k}(\mathcal{A})$	$1 + A(z) + A(z)^2 + \cdots + A(z)^k = \dfrac{1 - A(z)^{k+1}}{1 - A(z)}$
Sequences of length from ℓ to k	$\textsc{Seq}_{k \dots \ell}(\mathcal{A})$	$A(z)^\ell + A(z)^{\ell+1} + \cdots + A(z)^k = \dfrac{A^\ell - A(z)^{k+1}}{1 - A(z)}$

For example, the class and the ogf for the non-empty binary words of length at most 10 are

$$\mathcal{W} = \textsc{Seq}_{1 \dots 10}(\{0,1\}) \qquad \text{and} \qquad W(z) = \frac{2z - 2048z^{11}}{1 - 2z}.$$

3 Families of words

Fix a finite alphabet $\mathcal{A}$ where every letter is an atom (with unit size). The set of all **words on** $\mathcal{A}$ is simply the set sequences of letters made from this alphabet.

$$\mathcal{W} = \textsc{Seq}(\mathcal{A}) = A^* \qquad \text{and} \qquad W(z) = \frac{1}{1 - A(z)}$$

Let us return to binary words, but keep in mind how we might generalize all of this to a larger alphabet. Let us denote the atoms $\mathcal{Z}_0$ and $\mathcal{Z}_1$ as simply 0 and 1.

Example. Counting binary words the hard way
Consider the class $\mathcal{W}$ of binary words on $\mathcal{A} = \{0,1\}$. As above, we can write

$$\mathcal{W} = \textsc{Seq}(\mathcal{Z}_0 + \mathcal{Z}_1) = (0+1)^* = \{0,1\}^* \qquad \text{so} \qquad W(z) = \frac{1}{1 - 2z}.$$

But we can also decompose any such word by *cutting it just before each* 1. This results in a sequence of substrings called **blocks**. For example,

$$100110100 \;\mapsto\; |100|1|10|100 \;\mapsto\; (\epsilon, 100, 1, 10100)$$

The first block is a sequence of zeros (possibly of length zero). Each remaining block (if any) is a "1" followed by a sequence of zeros. We can recover the binary string from the sequence of blocks by concatenating them, such as

$$(00, 100, 10, 1, 1, 1) \mapsto 0010010111.$$

This decomposition is a combinatorial isomorphism, from which we may mechanically derive its ogf using the Product and Sequence constructions.

$$\mathcal{W} = \textsc{Seq}(\mathcal{Z}_0) \times \textsc{Seq}(\mathcal{Z}_1 \times \textsc{Seq}(\mathcal{Z}_0)) = 0^*(10^*)^*$$

We derive the OGF $W(z)$ from the atomic OGF's and by using the rules for "$\times$" and $\textsc{Seq}()$ as in the following table.

Combinatorial Class	Regular Expression	OGF
$\mathcal{Z}_0$	0	z
$\mathcal{Z}_1$	1	z
$\textsc{Seq}(\mathcal{Z}_0)$	0^*	$\frac{1}{1-z}$
$\mathcal{Z}_1 \times \textsc{Seq}(\mathcal{Z}_0)$	10^*	$z \cdot \frac{1}{1-z} = \frac{z}{1-z}$
$\textsc{Seq}(\mathcal{Z}_1 \times \textsc{Seq}(\mathcal{Z}_0))$	$(10^*)^*$	$\frac{1}{1-\frac{z}{1-z}}$
$\mathcal{W} = \textsc{Seq}(\mathcal{Z}_0) \times \textsc{Seq}(\mathcal{Z}_1 \times \textsc{Seq}(\mathcal{Z}_0))$	$0^*(10^*)^*$	$W(z) = \frac{1}{1-z} \cdot \frac{1}{1-\frac{z}{1-z}}$

It is reassuring to see that this expression for $W(z)$ simplifies algebraically to its more familiar form.

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

With the binomial theorem, we extract coefficients to count the binary words of length n.

$$W_n = [z^n]W(z) = [z^n]\frac{1}{1-2z} = 2^n.$$

This was a long-winded way to count the binary words of length n.

Unique representation: For other families of words, this type of decomposition of words into sequences of blocks can be very helpful for determining its ogf. *The biggest challenge is ensuring that each word is generated in a unique way.* For example, the combinatorial class

$$(0011 + 00 + 1)^*$$

generates the string 0011 in two different ways. Hence 0011 does not have a unique representation. Here we cannot count the words generated by $(0011+00+1)^*$ by translating to the generating functions using the above rules. In this example, there are exactly 5 different words of length four that can be generated, namely $\{0000, 0011, 1001, 1100, 1111\}$ but one can check with Maple that

$$[z^4]\frac{1}{1-(z+z^2+z^4)} = 6.$$

```
> series( 1/(1-(z+z^2+z^4)), z);
                    1 + z + 2 z^2 + 3 z^3 + 6 z^4 + 10 z^5 + O(z^6)
```

The string 0011 is counted twice by the generating function, once as (0011) and once as $(00, 1, 1)$.

We can fix this problem by deleting the string 0011 from the list of generators, since it is already generated as 00,1,1. The class of strings we want is, in fact, combinatorially equivalent to $(00 + 1)^*$.

Example. Consider the combinatorial class $\mathcal{F}$ of binary words specified by

$$(000 + 00001 + 10)^*$$

Is every word in $\mathcal{F}$ uniquely specified? If we write out a few words in $\mathcal{F}$, we are soon led to suspect that the answer is *yes*, But we need a proof! The best way to fashion a proof is to fashion a rule for breaking up any binary word w, where and reason that there is exactly one choice of how to do this at each step. The rule should either result in a sequence of parts from the set $\{000, 00001, 10\}$ to obtain w, or which gets stuck if w is not in $\mathcal{F}$. The presence of the word 000 in the list suggests that we can look at the number of zeros modulo 3 that appear in, say, the final block of zeros in w.

Observation. Let w be a binary word. Let k be the number of zeros in the last block of zeros in w (so $k = 0$ if w ends with "1").

1. If $k = 3\ell$ for some integer $\ell \geq 0$, then either
   (a) w contains no ones and w uniquely decomposes as $w = (000)^\ell$, or
   (b) w contains a one, and w uniquely decomposes as $w = w'(00001)(000)^\ell$ for some $w' \in \mathcal{F}$, or
   (c) $w \notin \mathcal{F}$.
2. If $k = 3\ell + 1$ for some integer $\ell \geq 0$, then either
   (a) w uniquely decomposes as $w = w'(10)(000)^\ell$ for some word $w' \in \mathcal{F}$, or
   (b) $w \notin \mathcal{F}$.
3. If $k = 3\ell + 2$ for some integer $\ell \geq 0$, then $w \notin \mathcal{F}$.

This leads to a simple recursive algorithm which outputs the only possible decomposition of w into the required blocks $w \in \mathcal{F}$. This proves that each word produced by $(000+00001+10)^*$ is uniquely specified. The generating function for $\mathcal{F}$ is obtained using the Sum, Product and Sequence constructions.

$$\mathcal{F} \cong \textsc{Seq}(\ \mathcal{Z}_0^3 + (\mathcal{Z}_0^4 \times \mathcal{Z}_1) + (\mathcal{Z}_1 \times \mathcal{Z}_0)\ ) \tag{1}$$

$$F(z) = \frac{1}{1 - (z^3 + z^5 + z^2)}.$$

We have expressed $F(z)$ as a rational function, so we can perhaps determine its counting sequence.

```
> F := 1/(1-(z^3+z^5+z^2));
  # First few terms:
    series(F, z, 9);
  # Number of strings of length 100
    'F[15]' = genfunc:-rgf_term(F,z,15);
  # Maple find general formula for F[n] fails because the denominator could not be factored.
    'F[n]' = genfunc:-rgf_expand(F,z,n);
```

$$F := \frac{1}{-z^5 - z^3 - z^2 + 1}$$

$$1 + z^2 + z^3 + z^4 + 3z^5 + 2z^6 + 5z^7 + 6z^8 + O(z^9)$$

$$F_{15} = 77$$

$$F_n = \sum_{\_R = RootOf(\_Z^5 + \_Z^3 + \_Z^2 - 1)} \left(- \frac{\left(\lim_{z \to \_R} \left(- \frac{z - \_R}{z^5 + z^3 + z^2 - 1} \right) \right) \left(\frac{1}{\_R} \right)^n}{\_R} \right)$$

Here, Maple failed to find a general formula for F_n because it could not factor the denominator of $F(z)$! Later we will see how we can find an approximate formula for F_n.

The Maple package `combstruct` can derive a generating function from the combinatorial specification in equation (1).

```
> # We use the combstruct package. Type ?combstruct for more info.
  with(combstruct):

  # We specify the combinatorial class (000+00001+10)* with the following set of equations.
  # Here Atom specifies an atom, and Epsion is a neutral element.
  Fclass := { Z0 = Atom, Z1 = Atom,
              Block1 = Prod(Z0,Z0,Z0),
              Block2 = Prod(Z0,Z0,Z0,Z0,Z1),
              Block3 = Prod(Z1,Z0),
              Fword = Sequence(Union(Block1,Block2,Block3)) }:

> # combstruct can count the Fclass words of length 15
  # and generate a random one (in an awkward format)
  'F[15]' = count([Fword,Fclass], size=15);
  randWord := draw([Fword,Fclass], size=15);
```

$$F_{15} = 77$$

$$randWord := Sequence(Prod(Z1, Z0), Prod(Z0, Z0, Z0), Prod(Z0, Z0, Z0, Z0, Z1), Prod(Z0, Z0, Z0, Z0, Z1))$$

```
> # Here is a procedure for making Fclass words prettier.
  cleanWord := word -> cat(eval(subs(Z0=0,Z1=1,Prod=(()->args),Sequence=(()->args),
                                     Epsilon=NULL, word))):
  cleanWord( randWord );
```

$$100000000100001$$

```
> # Here are the ordinary generating functions of classes in Fclass, and the first few terms of F(z)
  gfsolve( Fclass, labeled, z);
  Order:=9:   'F(z)' = gfseries( Fclass, 'labeled', z)[Fword(z)];
```

$$\left\{Block1(z) = z^3,\ Block2(z) = z^5,\ Block3(z) = z^2,\ Fword(z) = -\frac{1}{z^5+z^3+z^2-1},\ Z0(z) = z,\ Z1(z) = z\right\}$$

$$F(z) = 1 + z^2 + z^3 + z^4 + 3z^5 + 2z^6 + 5z^7 + 6z^8 + O(z^9)$$

Example. Let us find a construction for the combinatorial class $\mathcal{W}_{even}$ of binary words with the additional condition that **every block of 0s has of even length**. Thus, 0010000111001 and 111 are in the language but 00011 and 1001110 are not. A word is in $\mathcal{W}_{even}$ if and only if we can break it into parts, where each part is either '1' or '00'. Moreover, every word in $\mathcal{W}_{even}$ is uniquely generated this way. This gives the following combinatorial isomorphism and its ogf.

$$\mathcal{W}_{even} \cong (00+1)^* = \textsc{Seq}((\mathcal{Z}_0 \times \mathcal{Z}_0) + \mathcal{Z}_1) = \textsc{Seq}(\mathcal{Z}_0^2 + \mathcal{Z}_1).$$

$$W_{even}(z) = \frac{1}{1-(z^2+z)}$$

Maple gives first few terms of $W_{even}(z)$.

```
> series( 1/(1-(z+z^2)), z);
```

$$1 + z + 2z^2 + 3z^3 + 5z^4 + 8z^5 + O(z^6)$$

The counting sequence for $\mathcal{W}_{even}$ is the Fibonacci sequence. To see this directly, we notice that every word in $\mathcal{W}_{even}$ of size ≥ 2 is uniquely generated by appending either '1' or '00' to another word in $\mathcal{W}_{even}$. This gives the familiar recurrence

$$(W_{even})_n = (W_{even})_{n-1} + (W_{even})_{n-2}\,, \text{ for } n \geq 2.$$

A closed formula can be found by partial fractions. We factor the denominator

$$1 - z - z^2 = (1-\alpha z)(1-\beta z) = 1 - (\alpha+\beta)z + \alpha\beta z^2$$

Comparing coefficients, we have $1 = \alpha + \beta$ and $-1 = \alpha\beta = \alpha(1-\alpha)$. This solves to $\alpha, \beta = \frac{1\pm\sqrt{5}}{2}$. Here $\alpha = \frac{1+\sqrt{5}}{2}$ is the "golden ratio". Partial fractions gives

$$\frac{1}{(1-\alpha z)(1-\beta z)} = \frac{A}{1-\alpha z} + \frac{B}{1-\beta z}$$

$$1 + 0z = A(1-\beta z) + B(1-\alpha z) = (A+B) - (\beta A + \alpha B)z$$

Comparing coefficients, we have $1 = A + B$ and $0 = \beta A + \alpha B = \beta A + \alpha(1-A)$, which gives

$$A = \frac{\alpha}{\alpha-\beta} = \frac{1+\sqrt{5}}{2\cdot\sqrt{5}} = \frac{\sqrt{5}+5}{10} \qquad \text{and} \qquad B = 1 - \frac{\sqrt{5}+5}{10} = \frac{5-\sqrt{5}}{10}.$$

This results in a formula for the nth Fibonacci number.

$$\begin{aligned}[z^n]W_{even}(z) &= [z^n]\frac{A}{1-\alpha z} + [z^n]\frac{B}{1-\beta z}\\ &= A\alpha^n + B\beta^n\\ &= \frac{5+\sqrt{5}}{10}\left(\frac{1+\sqrt{5}}{2}\right)^n + \frac{5-\sqrt{5}}{10}\left(\frac{1-\sqrt{5}}{2}\right)^n\\ &\approx 0.723\cdot 1.62^n + 0.28\cdot(-0.62)^n.\end{aligned}$$

For $n \geq 0$, the second term in this sum has absolute value less than 1, so the n-th Fibonacci number equals the integer that is closest to

$$\frac{5+\sqrt{5}}{10}\left(\frac{1+\sqrt{5}}{2}\right)^n = \frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^{n+1}.$$

Example. We determine a construction for the combinatorial class $\mathcal{W}_{odd}$ of binary words with the additional condition that **every block of 0s has odd length**. This problem is worded a little vaguely; is the word 010001101 in $\mathcal{W}_{odd}$? In one sense, $10001101 \notin \mathcal{W}_{odd}$ since two consecutive 1s are separated by a block of length zero, and zero is not odd. Is the empty word in $\mathcal{W}_{odd}$? We have a few possible ways to define $\mathcal{W}_{odd}$ precisely.

Case 1: Every block of zeros (including blocks of length zero lying between two 1s) has odd length. Thus we exclude words that have two consecutive ones. We also exclude the empty word ϵ since ϵ is a block of zeros having length zero.

Case 2: This is like case 1, except we DO include ϵ.

Case 3: Every *nonempty* block of zeros has odd length, so words having consecutive 1s are included. The empty word is therefore included.

Let us count the shortest words in $\mathcal{W}_{odd}$, defined as in Case 1. (Case 2 is very similar. Case 3 will be homework.) Listing the words of length ≤ 4, we find

$$\mathcal{W}_{odd} = \{0, 1, 01, 10, 000, 010, 101, 0001, 1000, 0101, 1010, \dots\} \quad \text{so} \quad W(z) = 2z + 2z^2 + 3z^3 + 4z^4 + \dots$$

Suppose we cut each word $w \in \mathcal{W}_{odd}$ **immediately after each run of zeros**. If we make k cuts on w, then this divides w into exactly $k+1$ **blocks**. (If w ends with a zero, then we added a cut to the end of w, and we declare the last block to be the empty sequence ϵ.) We list the resulting sequence of blocks (including the empty block) for the above sample of words.

0	1	01	10	000	010	101	0001	1000	0101	1010	00000	10001	00010	01000	10101	01010
0 $\hat{}$	1	0 $\hat{}$ 1	10 $\hat{}$	000 $\hat{}$	0 $\hat{}$ 10 $\hat{}$	10 $\hat{}$ 1	000 $\hat{}$ 1	1000 $\hat{}$	0 $\hat{}$ 10 $\hat{}$ 1	10 $\hat{}$ 10 $\hat{}$	00000 $\hat{}$	1000 $\hat{}$ 1	000 $\hat{}$ 10	0 $\hat{}$ 1000	10 $\hat{}$ 10 $\hat{}$ 1	0 $\hat{}$ 10 $\hat{}$ 10 $\hat{}$
0 $\hat{}$ ϵ	1	0 $\hat{}$ 1	10 $\hat{}$ ϵ	000 $\hat{}$ ϵ	0 $\hat{}$ 10 $\hat{}$ ϵ	10 $\hat{}$ 1	000 $\hat{}$ 1	1000 $\hat{}$ ϵ	0 $\hat{}$ 10 $\hat{}$ 1	10 $\hat{}$ 10 $\hat{}$ ϵ	00000 $\hat{}$ ϵ	1000 $\hat{}$ 1	000 $\hat{}$ 10	0 $\hat{}$ 1000	10 $\hat{}$ 10 $\hat{}$ 1	0 $\hat{}$ 10 $\hat{}$ 10 $\hat{}$ ϵ

For each $w \in \mathcal{W}_{odd}$ this procedure produces a unique sequence of non-empty *blocks* from which we can uniquely reconstruct the word. If there are at least two block, then we have coloured the first block red, the last block green. Every remaining block (if any) is a *middle* block and is coloured blue. For example

$$10010001000000010100 \longleftrightarrow (100, 1000, 10000000, 10, 100, \epsilon)$$

Note: There is one exceptional word, $1 \in \mathcal{W}_{odd}$, with no cuts and just one block which we will have to countseparately.

What can a block b_i of a word

$$w = b_1\ b_2\ b_3\ \ldots\ b_{k-1}\ b_k\ \in \mathcal{W}_{odd}$$

look like? This depends on whether b_i is the *first* block b_1, the *last* block b_k, or a *middle* block b_i $(1 < i < k)$.

Every middle block b_i $(1 < i < k)$ is a 1 followed by an odd number of zeros, so b_i is a member of the combinatorial class $\mathcal{M}$ where

$$\mathcal{M} = 10(00)^*\,.$$

The first block b_1 allows another possibility. If w has at least two blocks and the first digit of w is 1, then b_1 also belongs to $\mathcal{M}$; if w starts with 0 then b_1 must be an odd-length block of zeros. That is, if $k \geq 2$, then $b_1 \in \mathcal{F}$ where

$$\mathcal{F} = 10(00)^* + 0(00)^*.$$

The last block b_k also allows two possibilities. If the last digit of w is 0, then b_k is the empty string ϵ; otherwise b_k is the single digit 1. That is, if $k \geq 2$ then $b_k \in \mathcal{L}$ where

$$\mathcal{L} = \epsilon + 1.$$

We have shown that every word in w which has at least two blocks belongs to the class

$$\mathcal{F}\mathcal{M}^*\mathcal{L} = (10(00)^* + 0(00)^*)\ \ (10(00)^*)^*\ (10(00)^* + 1).$$

We still need to describe words w having just one block (so $k = 1$ and $w = b_1$). If w ends with a 0, then $k \geq 2$. So if $k = 1$, then $w = 1$ or $w =$. Since $\mathcal{W}_{odd}$ does not contain the empty word, the word $w = 1$ is the only word in $\mathcal{W}_{odd}$ that is not a member of $\mathcal{F}\mathcal{M}^*\mathcal{L}$.

$$\mathcal{W}_{odd}\ \subseteq\ (10(00)^* + 0(00)^*)\ \ (10(00)^*)^*\ (10(00)^* + 1)\ +\ 1. \tag{2}$$

To show that this is a combinatorial isomorphism, we still need to verify the inverse containment "$\supseteq$". This is not difficult; we need only check that every nonempty block of zeros of every word in the right hand side of (2) has odd length. We may now find the OGF $W_{odd}(z)$ in the usual way.

$$\begin{aligned}
\mathcal{W}_{odd} \cong\ & (10(00)^* + 0(00)^*) && (10(00)^*)^* && (10(00)^* + 1) && +1 \\
& \updownarrow && \updownarrow && \updownarrow && \updownarrow \\
W_{odd}(z) =\ & \left(\tfrac{z^2}{1-z^2} + \tfrac{z}{1-z^2}\right) && \tfrac{1}{1-\frac{z^2}{1-z^2}} && \left(\tfrac{z^2}{1-z^2} + z\right) && +z \\
=\ & \ldots && && && \text{(after some tedious work)} \\
=\ & \frac{2z + 2z^2 - z^3}{1 - 2z^2} && && &&
\end{aligned}$$

Case 1 simplified:
We can sometimes derive a slightly simpler description of $\mathcal{W}_{odd}$ by allowing more cuts in the decomposition. In this case, we can make a cut at every location which is **either just after a block of 0 or just before each digit 1**. This is almost the same rule as above, except for words that start with "1" get one more cut at the start, such as $1000101000001 \longrightarrow \hat{}1000\hat{}10\hat{}100000\hat{}1$..

Now it is possible that the first and last blocks of $w = b_1\ b_2\ b_3\ \ldots\ b_{k-1}\ b_k$ are the empty string ϵ. For example,

$$10010001000000010100 \longleftrightarrow (\epsilon, 100, 1000, 10000000, 10, 100, \epsilon)$$

Suppose that a word w decomposes into at least two blocks, so $k \geq 0$. The first block b_1 must either be ϵ or an odd block of zeros. The class of possible first blocks

$$\mathcal{F} = \{b_1 \mid w = b_1\ b_2\ b_3\ \ldots\ b_{k-1}\ b_k, w \in \mathcal{W}, k \geq 1\}$$

is a combinatorial class specified by

$$\mathcal{F} = \epsilon + 0(00)^*$$

As before, each middle block b_i, $(2 \le i \le k-1)$ is a "1" followed by an odd sequence of zeros. The set of middle blocks forms the combinatorial class $\mathcal{M}$ specified by

$$\mathcal{M} = 10(00)^*$$

Again the last block b_k is either ϵ or the single digit "1". They form the combinatorial class specified by

$$\mathcal{L} = (\epsilon + 1)$$

The advantage of this specification is that the word $1 \in \mathcal{W}_{odd}$ now belongs to $\mathcal{F}\mathcal{M}^*\mathcal{L}$. The disadvantage is that the empty word also belongs to $\mathcal{F}\mathcal{M}^*\mathcal{L}$, but ϵ is not a member of $\mathcal{W}_{odd}$.

Again every word described by the right hand side of (3) (except for ϵ) belongs to $\mathcal{W}_{odd}$. This give us

$$\mathcal{W}_{odd} + \epsilon \;\cong\; \mathcal{F}\mathcal{M}^*\mathcal{L} \;=\; (\epsilon + 0(00)^*)\;(10(00)^*)^*\,(\epsilon + 1). \tag{3}$$

(So the right hand side of (3) describes Case 2, not Case 1.)

We now have

$$\begin{aligned} W_{odd}(z) &= \left(1 + \frac{z}{1-z^2}\right) \frac{1}{1 - \frac{z^2}{1-z^2}} (1+z) \;-\; 1 \\ &= \ldots \qquad \text{(after not quite as much work as before)} \\ &= \frac{2z + 2z^2 - z^3}{1 - 2z^2} \end{aligned}$$

General approach for describing classes of words

If we need to find a construction for a class $\mathcal{W}$ of words where there are restrictions on the allowed blocks of digits, we can often take following steps.

- Describe a procedure for breaking a word up into a sequence $b_1 b_2 \ldots b_k$ of *blocks*. Often the description of a block depends on whether or not it is the first or last block. It is often a good idea to allow the first or the last block to be the empty string ϵ, as this usually leads to a simpler description.
- Find a combinatorial description of each middle block b_i, $(1 < i < k)$. Let $\mathcal{M}$ be the combinatorial class describing each middle block.
- Find a description of the class $\mathcal{F}$ of possible first blocks b_1. Typically b_1 will either be ϵ or will take an alternative form that is a bit simpler than $\mathcal{M}$ was. That is, we usually find that $\mathcal{F} = \mathcal{E} + \mathcal{M}'$ where $\mathcal{M}'$ is an alternate, slightly simpler form of $\mathcal{M}$.
- Find a description of the class $\mathcal{L}$ of possible last blocks. Again we can usually describe $\mathcal{L} = \mathcal{E} + \mathcal{M}''$ where $\mathcal{M}'$ is an alternate, slightly simpler form of $\mathcal{M}$.
- At this point, you have shown that $\mathcal{W} \subseteq \mathcal{F}\mathcal{M}^*\mathcal{L}$. Now check that every word in $\mathcal{F}\mathcal{M}^*\mathcal{L}$ belongs to $\mathcal{W}$, and take heed of whether $\epsilon \in \mathcal{W}$. Assuming $\epsilon \in \mathcal{W}$, we now have

  $$\mathcal{W} \cong \mathcal{F}\;\mathcal{M}^*\;\mathcal{L}, \quad \text{where typically } \mathcal{F} = \mathcal{E} + \mathcal{M}' \text{ and } \mathcal{L} = \mathcal{E} + \mathcal{M}''.$$

  This description includes the empty word ϵ, so we might have to subtract the neutral class $\mathcal{E}$ from the right hand side.

Exercise. Find a closed formula for $(W_{odd})_n$, the number of words of length n in $\mathcal{W}_{odd}$ (with the definition of Case 1).

Solution:

A quick way to extract coefficients of $W_{odd}(z) = \dfrac{2z + 2z^2 - z^3}{1 - 2z^2}$ is to substitute $y = z^2$ into its denominator, and observe that $[z^{2k}]f(z^2) = [y^k]f(y)$.

$$[z^n]\frac{1}{1-2z^2} = \begin{cases} [y^k]\frac{1}{1-2y} = 2^k = 2^{n/2} & \text{if } n = 2k \text{ is even and } n \geq 0 \\ 0 & \text{otherwise.} \end{cases}$$

TApplying the linearity rule, sum rule and reduction rule,

$$[z^n]\frac{2z + 2z^2 - z^3}{1 - 2z^2} = [z^{n-1}]\frac{2}{1-2z^2} + [z^{n-2}]\frac{2}{1-2z^2} - [z^{n-3}]\frac{1}{1-2z^2}$$

$$= \begin{cases} 0 & = 0 & \text{if } n = 0 \\ 2 \cdot 2^{(n-1)/2} & = 2 & \text{if } n = 1 \\ 0 + 2 \cdot 2^{(n-2)/2} - 0 & = 2^{n/2} & \text{if } n \geq 2 \text{ is even} \\ 2 \cdot 2^{(n-1)/2} + 0 - 2^{(n-3)/2} & = 3 \cdot 2^{(n-3)/2} & \text{if } n \geq 3 \text{ is odd.} \end{cases}$$

This agrees with the above partial listing of $\mathcal{W}_{odd}$ found with Maple.
We can also find the coefficients of $W_{odd}(z)$ using partial fractions, by factoring its denominator as

$$1 - 2z^2 = (1 - \sqrt{2}\,z)(1 + \sqrt{2}\,z).$$

This just a little tedious to do by hand, but we can also use Maple.

```
> with(genfunc):
> # OGF for nonempty binary words where every (nonempty) block of zeros has odd length

  W := (2*z+2*z^2-z^3)/(1-2*z^2);
  series( W, z,10);
```

$$W := \frac{-z^3 + 2z^2 + 2z}{-2z^2 + 1}$$

$$2z + 2z^2 + 3z^3 + 4z^4 + 6z^5 + 8z^6 + 12z^7 + 16z^8 + 24z^9 + O(z^{10})$$

```
> # General formula for number of n-bit words in W
  # It is valid for n>1

  'W[n]' = genfunc:-rgf_expand( W, z, n);
  seq('W'[k]=simplify(eval(rhs(%), n=k)), k=0..3);  # General formula fails for small n
  rgf_sequence('boundary', W, z, WW);               # Correct values for small n
```

$$W_n = \frac{(3\sqrt{2}+4)(\sqrt{2})^n}{8} - \frac{(3\sqrt{2}-4)(-\sqrt{2})^n}{8}$$

$$W_0 = 1, W_1 = \frac{3}{2}, W_2 = 2, W_3 = 3$$

$$WW(1) = 2, WW(2) = 2, WW(3) = 3$$

$$(W_{odd})_n = \begin{cases} 0 & \text{if } n = 0 \\ 2 & \text{if } n = 1 \\ \left(\frac{1}{2} + \frac{3}{8}\sqrt{2}\right)2^{n/2} + \left(\frac{1}{2} - \frac{3}{8}\sqrt{2}\right)(-2)^{n/2} & \text{if } n \geq 2 \end{cases}$$