DE GRUYTER
OPEN

PU. M. A. Vol. **25** (2015), No.1, pp. 30 – 44

A generating function for bit strings with no Grand Dyck pattern matching

Antonio Bernini
Dipartimento di Matematica e Informatica "U. Dini"
Università degli Studi di Firenze
Viale G.B. Morgagni 65, 50134 Firenze, Italy
email: antonio.bernini@unifi.it

and
Stefano Bilotta
Dipartimento di Matematica e Informatica "U. Dini"
Università degli Studi di Firenze
Viale G.B. Morgagni 65, 50134 Firenze, Italy
email: stefano.bilotta@unifi.it

and
Elisa Pergola
Dipartimento di Matematica e Informatica "U. Dini"
Università degli Studi di Firenze
Viale G.B. Morgagni 65, 50134 Firenze, Italy
email: elisa.pergola@unifi.it

and
Renzo Pinzani
Dipartimento di Matematica e Informatica "U. Dini"
Università degli Studi di Firenze
Viale G.B. Morgagni 65, 50134 Firenze, Italy
email: renzo.pinzani@unifi.it

(Received: September 22, 2014, and in revised form June 8, 2015)

Abstract. We study the construction and the enumeration of bit strings, or binary words in $\{0,1\}^*$, having more 1's than 0's and avoiding a set of Grand Dyck patterns which form a cross-bifix-free set. We give a particular *jumping and marked succession rule* which describes the growth of such words according to the number of 1's. Then, we give the enumeration of the class by means of generating function.

Mathematics Subject Classification(2010). 68Q45, 68R15.

Keywords: binary word, pattern avoidance, Grand Dyck pattern.

DOI: 10.1515/puma-2015-0003

1 Introduction

The problem of the occurrence of a fixed *pattern* in long sequences of observation is interesting in many scientific problems.

For example, in the area of computer networks security, the detection of intrusions, which are becoming increasingly frequent, is very important. Intrusion detection is primarily concerned with the detection of illegal activities and acquisitions of privileges that cannot be checked with information flow and access control models. There are several approaches to intrusion detection, but recently this subject has been studied in relation to pattern matching (see [1, 11, 14]).

Moreover, in the area of computational biology it is interesting to control the occurrences of a particular pattern in a genomic sequence over the alphabet $\{A, G, C, T\}$, see for instance [16, 18].

These applications are concerned with both the enumeration and the construction of particular words avoiding a given pattern over an alphabet.

In the set of binary words, the ones avoiding a fixed pattern can be described by a regular language and can be enumerated by using classical results leading to rational generating functions (see, e.g., [12, 13, 17]). If the set is restricted to words with no more 0's than 1's, then the language for words avoiding a fixed pattern is more difficult to deal with. For each forbidden pattern an "ad hoc" grammar (from which the generating function can be obtained) should be defined. Consequently, for each pattern a different generating function enumerating the words avoiding it should be computed.

The aim of the present work is to consider an unified algorithmic approach by means of particular succession rules, studying the construction and the enumeration of binary words in $\{0, 1\}^*$ having more 1's than 0's and avoiding a particular set of patterns.

This general approach, which is used in some previous works (see for instance [4, 6]), requires the forbidden set of consecutive patterns to be a cross-bifix-free set, that is, a set of words such that no prefix of any word is a suffix of any other word (see [5, 7]).

In [4] the authors deal with the binary words having more 1's than 0's and avoiding a set of primitive Dyck words. In the present work we extend the result in [4] in order to avoid a set of Grand Dyck words and we show that if the set of forbidden Grand Dyck words is a cross-bifix-free set, then the number of words in $\{0, 1\}^*$ having more 1's than 0's and avoiding such a set does not depend on the shape of the avoided patterns themselves, but only on the total number of ones in the patterns.

In the following section we give some basic definitions and notations related to the concepts of succession rule, cross-bifix-free sets and Grand Dyck patterns. In Section 3 we present a combinatorial interpretation of the succession rule describing the growth of the class of binary words having more 1's than 0's and avoiding a cross-bifix-free set of Grand Dyck patterns. In particular, the problem of associating a word to a path in the generating tree obtained by the succession rule is solved by introducing an algorithm which constructs all binary words having more 1's than 0's and then excludes those ones containing the forbidden patterns. Finally, in Section 4 a generating function is given by using the ECO-method [2, 3] for the enumeration of combinatorial objects which admit recursive descriptions in terms of generating trees.

2 Basic definitions and notations

We introduce two essential concepts for the further development of the work.

2.1 Succession rules

A *succession rule* Ω on a set $\Sigma \subseteq \mathbb{N}$ is a system constituted by an *axiom* (a), with $a \in \Sigma$, and a set of *productions* of the form:

$$\{(k) \overset{1}{\rightsquigarrow} (e_1(k))(e_2(k)) \dots (e_k(k))\}_{k \in \Sigma} \;\; e_i(k) \in \Sigma,\, 1 \leq i \leq k.$$

A production constructs, for any given k, its k *successors* labelled by $(e_1(k)), (e_2(k)), \dots, (e_k(k))$. In most of the cases, for a succession rule Ω, we use the more compact notation:

$$\Omega : \left\{ \begin{array}{ll} (a) & \\ (k) & \overset{1}{\rightsquigarrow} (e_1(k))(e_2(k)) \dots (e_k(k)). \end{array} \right. \tag{1}$$

In this context Σ is called the set of *labels.* The rule Ω can be represented by means of a *generating tree*, that is an infinite tree with the root labelled by (a). Each node labelled by (k) has k successors labelled $(e_1(k)), (e_2(k)), \dots, (e_k(k))$, respectively. By convention the root is at level 0, and a node lies at level n if its parent lies at level $n-1$. A tree is a generating tree for a class of combinatorial objects if there exists a bijection between the objects of size n and the nodes at level n in the tree, and in such a case a given object can be coded by the sequence of labels met from the root of the generating tree to the object itself.

For instance, the succession rule for Catalan numbers is

$$\left\{ \begin{array}{ll} (2) & \\ (k) & \overset{1}{\rightsquigarrow} (2)(3) \dots (k)(k+1) \end{array} \right. \tag{2}$$

and some levels of its generating tree are shown in Figure 1. We refer to [3] for further details and examples on succession rules.

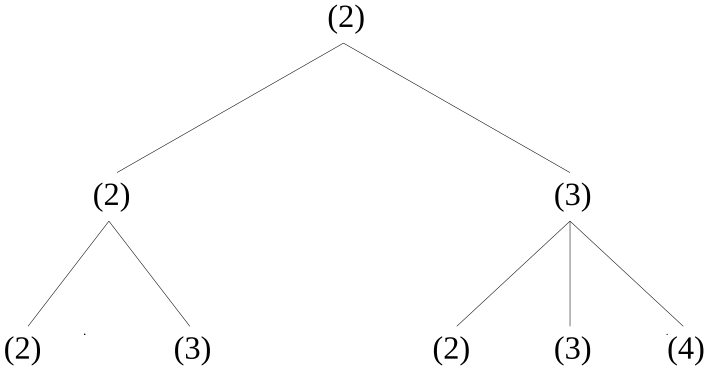

Figure 1: Some levels of the generating tree associated with the succession rule (2).

We remark that, from the above definition, a node labelled by (k) has precisely k successors. However, we can also consider succession rules, introduced in [9], in which the value of a label does not necessarily represent the number of its successors. As an instance see the succession rule (3) whose generating tree is shown in Figure 2.

$$\begin{cases} (0) & \\ (k) & \overset{1}{\rightsquigarrow} (k+1)(k)\dots(1)(0). \end{cases} \tag{3}$$

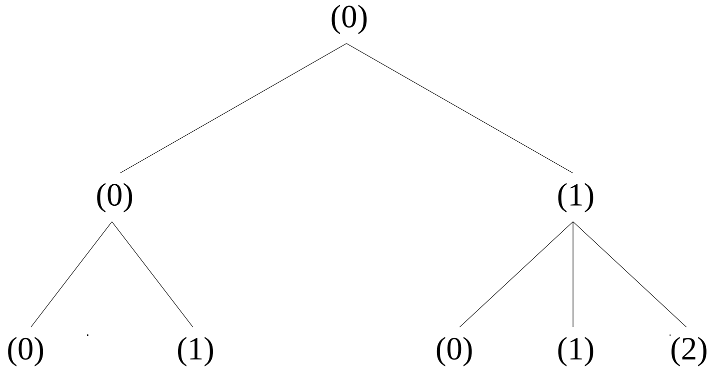

Figure 2: Some levels of the generating tree associated with the succession rule (3).

Succession rules such as (1) or (3) are not sufficient to handle all enumeration problems and so we consider a slight generalization called *jumping succession rule*. Roughly speaking, the idea is to consider a set of productions acting on the objects of a class and producing successors at different levels. To indicate a jumping succession rule Γ is used the following notation:

$$\Gamma : \begin{cases} (a) & \\ (k) & \overset{1}{\rightsquigarrow} (e_1(k))(e_2(k))\dots(e_k(k)) \\ (k) & \overset{j}{\rightsquigarrow} (d_1(k))(d_2(k))\dots(d_k(k)). \end{cases} \tag{4}$$

The tree associated with Γ has the property that each node labelled by (k) and lying at level n produces two sets of successors, the first set at level $n+1$ and having labels $(e_1(k)), (e_2(k)), \dots, (e_k(k))$, respectively, and the second one at level $n+j$, with $j > 1$, and having labels $(d_1(k)), (d_2(k)), \dots, (d_k(k))$, respectively. For example, the jumping succession rule (5) counts the number of *2-generalized Motzkin paths* and Figure 3 shows some levels of the associated generating tree. For more details about these topics, see [10].

$$\begin{cases} (1) & \\ (k) & \overset{1}{\rightsquigarrow} (1)(2)\cdots(k-1)(k+1) \\ (k) & \overset{2}{\rightsquigarrow} (k). \end{cases} \tag{5}$$

Another generalization is used in [6], where the authors deal with *jumping and marked succession rules*. In this case the labels appearing in a jumping succession rule can be marked, and the *marked* labels are considered together with the unmarked labels.

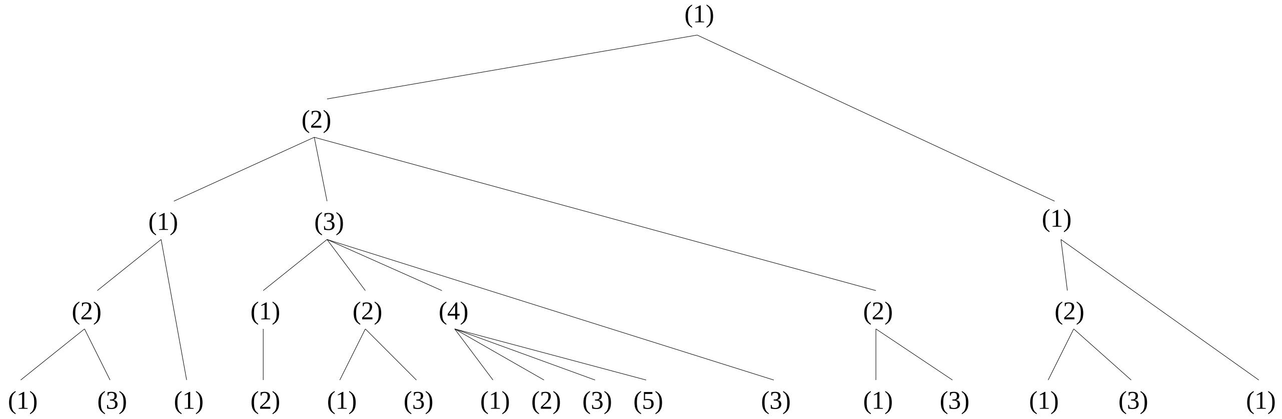

Figure 3: Four levels of the generating tree associated with the succession rule (5).

A *jumping and marked generating tree* is a rooted labelled tree where there appear marked and unmarked labels according to the corresponding succession rule. The main property is that in the generating tree a marked label $(\overline{e})$ kills or annihilates the unmarked label (e) lying on the same level n. In particular, the enumeration of the combinatorial objects in a class is the difference between the number of unmarked and marked labels lying on a given level.

Note that, the compact notation:

$$\begin{cases} (a) & \\ (k) & \overset{1}{\rightsquigarrow} (e_1(k))(e_2(k))\dots(e_k(k)) \\ (k) & \overset{j}{\rightsquigarrow} (\overline{d_1(k)})(\overline{d_2(k)})\dots(\overline{d_k(k)}) \end{cases} \tag{6}$$

describes also the derivation for $(\overline{k})$:

$$\begin{cases} (\overline{k}) & \overset{1}{\rightsquigarrow} (\overline{e_1(k)})(\overline{e_2(k)})\dots(\overline{e_k(k)}) \\ (\overline{k}) & \overset{j}{\rightsquigarrow} (d_1(k))(d_2(k))\dots(d_k(k)) \end{cases} \tag{7}$$

and for any label (k), we have $(\overline{\overline{k}}) = (k)$.

By the way, each succession rule (1) can be trivially rewritten as (8)

$$\begin{cases} (a) & \\ (k) & \overset{1}{\rightsquigarrow} (e_1(k))(e_2(k))\dots(e_k(k))(k) \\ (k) & \overset{1}{\rightsquigarrow} (\overline{k}). \end{cases} \tag{8}$$

For example, the succession rule for Catalan numbers can be rewritten in the form (9) and Figure 4 shows some levels of the associated generating tree.

$$\begin{cases} (2) & \\ (k) & \overset{1}{\rightsquigarrow} (2)(3)\dots(k)(k+1)(k) \\ (k) & \overset{1}{\rightsquigarrow} (\overline{k}). \end{cases} \tag{9}$$

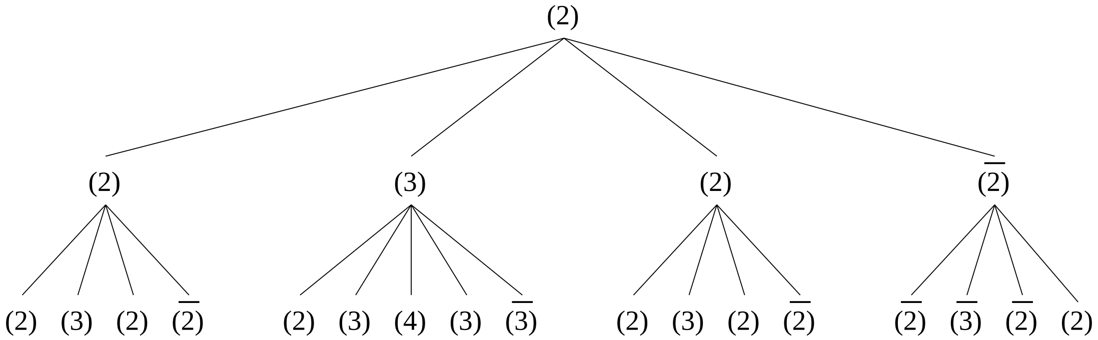
(2)
(2) (3) (2) $(\overline{2})$
(2) (3) (2) $(\overline{2})$ (2) (3) (4) (3) $(\overline{3})$ (2) (3) (2) $(\overline{2})$ $(\overline{2})$ $(\overline{3})$ $(\overline{2})$ (2)

Figure 4: Three levels of the generating tree associated with the succession rule (9).

2.2 Cross-bifix-free sets

Let Σ be a finite non-empty set. Let $\omega = uwv$ be a word in Σ^*, then u is called *prefix* of ω and v is called *suffix* of ω. A *bifix* of ω is a subsequence of ω that is both its prefix and suffix.

A word ω of Σ^* is said to be *bifix-free* [15] if and only if no strict prefix of ω is also a suffix of ω. Therefore, ω is bifix-free if and only if $\omega \neq uwu$, being u any necessarily non-empty word and w any word. Obviously, a necessary condition for ω to be bifix-free is that the first and the last letter of ω must be different.

In the monoid $\{0,1\}^*$, the word 111010100 of length $n = 9$ is bifix-free, while the word 101001010 contains two bifixes, 10 and 1010.

Let $BF_q(n)$ denote the set of all bifix-free words of length n over an alphabet of fixed size q. Let $q > 1$ and $n > 1$ be fixed. Two distinct words $\omega, \omega' \in BF_q(n)$ are said to be *cross-bifix-free* if and only if no strict prefix of ω is also a suffix of ω' and vice-versa. “Cross-bifix-free” is often “nonoverlapping” in the literature.

The binary words 111010100 and 110101010 in $BF_2(9)$ are cross-bifix-free, while the binary words 111001100 and 110011010 in $BF_2(9)$ have the cross-bifix 1100.

A subset of $BF_q(n)$ is said to be *cross-bifix-free set* if and only if for each ω, ω', with $\omega \neq \omega'$, in this set, ω and ω' are cross-bifix-free.

Let $F \subset \{0,1\}^*$ be the class of binary words ω such that $|\omega|_0 \leq |\omega|_1$ for any $\omega \in F$, where $|\omega|_0$ and $|\omega|_1$ denote the number of 0’s and 1’s in ω, respectively. Each word $\omega \in F$ can be naturally represented as a lattice path on the Cartesian plane by associating a *rise step*, defined by $(1,1)$ and denoted by x, to each 1 in F, and a *fall step*, defined by $(1,-1)$ and denoted by $\overline{x}$, to each 0 in F. From here on, we refer interchangeably to words or their graphical representations on the Cartesian plane, that is paths. The definition of bifix-free and cross-bifix-free can be easily extended to paths.

A lattice path on the Cartesian plane using the steps $(1,1)$ and $(1,-1)$ and running from $(0,0)$ to $(2m,0)$, with $m \geq 0$, is said to be *Grand-Dyck* or *Binomial* path of length $2m$ (see [8] for further details).

In this paper we are interested in studying the subclass $F^{[\mathcal{P}]}$ of F of binary words excluding a set $\mathcal{P} = \{\mathfrak{p}_1, \mathfrak{p}_2, \cdots, \mathfrak{p}_m\}$ of Grand Dyck patterns such that $\mathfrak{p}_i = p_{i,0}p_{i,1}\ldots p_{i,h_i-1} \in \{0,1\}^{h_i}$, where $p_{i,k}$ is the k-th letter of $\mathfrak{p}_i$, i.e. a word ω in $F^{[\mathcal{P}]}$ does not admit a sequence of consecutive indices

$j, j+1, \ldots, j+h_i-1$ such that $w_j w_{j+1} \ldots w_{j+h_i-1} = p_{i,0} p_{i,1} \ldots p_{i,h_i-1}$, for any i, $1 \leq i \leq m$.

3 The generating algorithm for the class $F^{[\mathcal{P}]}$

In this section, we define an algorithm to construct the set $F^{[\mathcal{P}]}$ where $\mathcal{P} = \{\mathfrak{p}_1, \mathfrak{p}_2, \cdots, \mathfrak{p}_m\} = \{\mathfrak{p}_1(j_1), \mathfrak{p}_2(j_2), \ldots, \mathfrak{p}_m(j_m)\}$ is a cross-bifix-free set of Grand Dyck patterns, none include in any other, such that $|\mathfrak{p}_l(j_l)|_1 = |\mathfrak{p}_l(j_l)|_0 = j_l$, $1 \leq l \leq m$.

The growth of the set $F^{[\mathcal{P}]}$, according to the number of rise steps or equivalently the number of ones, can be described by the following jumping and marked succession rule:

$$\begin{cases} (0) \\ (k) \overset{1}{\rightsquigarrow} (k+1)(k)\cdots(1)(0_2)(0_1) & k \geq 0 \\ (0) \overset{j_1}{\rightsquigarrow} (\overline{0_2}) \\ (k) \overset{j_1}{\rightsquigarrow} (\overline{k})(\overline{k-1})\cdots(\overline{1})(\overline{0_2})(\overline{0_1}) & k \geq 1 \\ (0) \overset{j_2}{\rightsquigarrow} (\overline{0_2}) \\ (k) \overset{j_2}{\rightsquigarrow} (\overline{k})(\overline{k-1})\cdots(\overline{1})(\overline{0_2})(\overline{0_1}) & k \geq 1 \\ \vdots \\ (0) \overset{j_m}{\rightsquigarrow} (\overline{0_2}) \\ (k) \overset{j_m}{\rightsquigarrow} (\overline{k})(\overline{k-1})\cdots(\overline{1})(\overline{0_2})(\overline{0_1}) & k \geq 1. \end{cases} \tag{10}$$

This rule can be represented as a tree having its root labelled (0) and each node with label (k) at level n gives $k+3$ sons at level $n+1$ labelled $(k+1), \cdots, (1), (0_2), (0_1)$ and $k+2$ sons at level $n+j_l$, $1 \leq l \leq m$, with labels $(\overline{k}), \cdots, (\overline{1}), (\overline{0_2}), (\overline{0_1})$, if $k \geq 1$, or only one son with label $(\overline{0_2})$ at level $n+j_l$, $1 \leq l \leq m$, if $k=0$. The subscripts of labels (0) are simply used in order to distinguish the two labels one from each other, since they are obtained in two different ways in the generating process. Note that the labels (0_1) and (0_2) have the same set of successors regardless their subscripts.

The generating algorithm associates a lattice path in $F^{[\mathcal{P}]}$ to a sequence of labels obtained by means of the succession rule (10). This give a construction for the set $F^{[\mathcal{P}]}$ according to the number of rise steps or equivalently the number of ones.

The axiom (0) is associated to the empty path ε.
A path $\omega \in F$, with n rise steps and such that its endpoint has ordinate k, generates $k+3$ paths with $n+1$ rise steps, according to the first production of (10) having $k+1, \ldots, 1, 0, 0$ as endpoint ordinates, respectively.

The first $k+2$ paths are obtained by adding to ω a sequence of steps consisting of one rise step followed by $k+1-h$, $0 \leq h \leq k+1$, fall steps, see Figure 5. Each path so obtained has the property that its rightmost suffix beginning from the x-axis, either remains strictly above the x-axis itself or ends on the x-axis by a fall step. Note that in this way the paths ending on the x-axis and having a rise step as last step are never obtained. These paths have the label (0_1) of the first production in (10) and the way to obtain them will be described in the sequel.

Let us denote by $L_l = (x_{L_l}, y_{L_l})$ and $R_l = (x_{R_l}, y_{R_l})$ the initial and last point of a pattern $\mathfrak{p}_l$, respectively. We define a *marked forbidden pattern* $\mathfrak{p}_l$ as a pattern $\mathfrak{p}_l \in \mathcal{P}$ whose steps cannot be split, that is they must always be contained all together in that defined sequence. We say that a point is

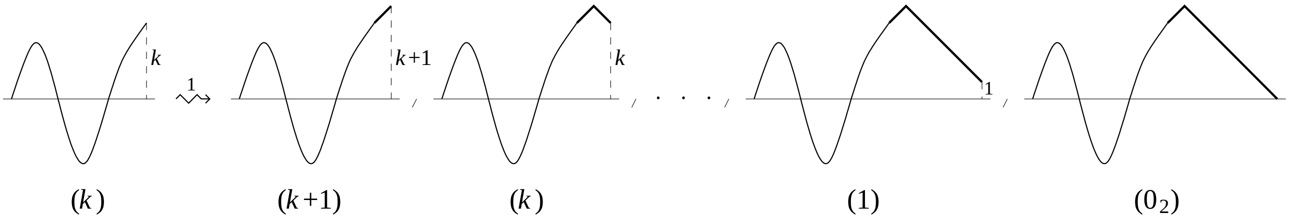

Figure 5: The mapping associated to $(k) \overset{1}{\rightsquigarrow} (k+1)(k)\ldots(1)(0_2)$ of (10).

strictly contained in a given marked forbidden pattern $\mathfrak{p}_l$ if it is in $\mathfrak{p}_l$ and it is different from both L_l and R_l.

We denote a marked forbidden pattern $\mathfrak{p}_l$ by drawing its minimal bounding rectangle B_l. A rectangle B_l is like a *black box*, in the sense that it masks the included pattern $\mathfrak{p}_l$ (see Figure 6). Consequently, we have two forms of a same forbidden pattern: a marked forbidden pattern and a no marked one, in accordance with the meaning of a jumping and marked succession rule.

A cut operation, i.e the procedure which splits a given path into two subpaths, is not possible within a marked forbidden pattern $\mathfrak{p}_l$. After a cut operation, it is not allowed to switch any rise step with a fall one, and viceversa, inside a marked forbidden pattern, but it can be translated.

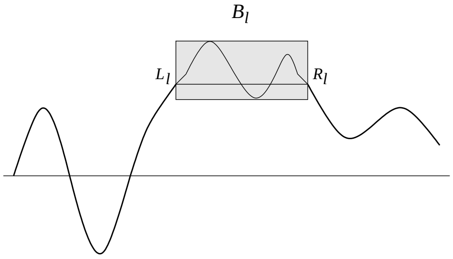

Figure 6: A marked forbidden pattern.

A path $\omega \in F$, with n rise steps and such that its endpoint has ordinate 0, provides one path, with $n + j_l$ rise steps, $1 \leq l \leq m$, according to the production $(0) \overset{j_l}{\rightsquigarrow} (\overline{0_2})$ of (10). The obtained path has 0 as endpoint ordinate and it is obtained by adding to ω a sequence of steps consisting of the marked forbidden pattern $\mathfrak{p}_\mathfrak{l}$ in bounding rectangle B_l, see Figure 7.

A path $\omega \in F$, with n rise steps and such that its endpoint has ordinate $k \geq 1$, provides $k+2$ paths, with $n + j_l$ rise steps, $1 \leq l \leq m$, according to the production $(k) \overset{j_l}{\rightsquigarrow} (\overline{k})(\overline{k-1})\cdots(\overline{1})(\overline{0_2})(\overline{0_1})$ of (10), having $k, \ldots, 1, 0, 0$ as endpoint ordinate, respectively. The first $k+1$ labels are obtained by adding to ω a sequence of steps consisting of the marked forbidden pattern $\mathfrak{p}_\mathfrak{l}$ followed by $k - h$, $0 \leq h \leq k$, fall steps, see Figure 8.

At this point the label (0_1) due to the productions of (10) is associated with paths which either do not contain marked forbidden patterns in its rightmost suffix and end on the x-axis by a rise step or having the initial point L_l in the rightmost marked forbidden pattern at ordinate less than 0.

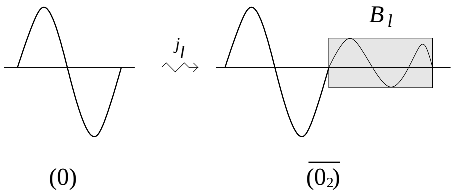

Figure 7: The mapping associated to $(0) \overset{j_l}{\rightsquigarrow} (\overline{0_2})$ of (10).

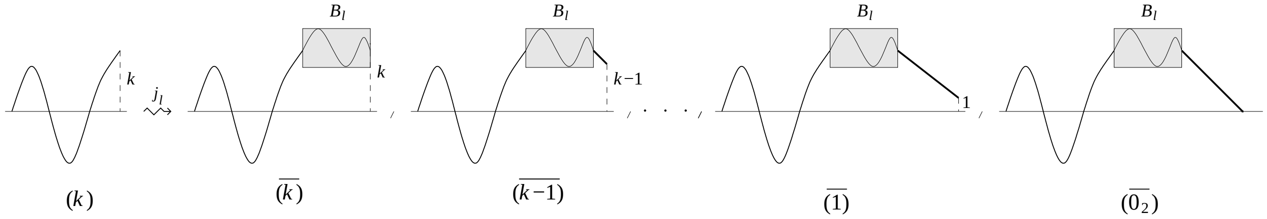

Figure 8: The mapping associated to $(k) \overset{j_l}{\rightsquigarrow} (\overline{k}) \ldots (\overline{1})(\overline{0_2})$ of (10).

In order to obtain the path labelled by (0_1) according to the first production of (10), we consider the path ω' obtained from ω by adding a sequence of steps consisting of one rise step followed by k fall steps. In order to obtain the path labelled by (0_1) according to every one of the other productions of (10), we consider the paths ω' obtained from ω by adding a sequence of steps consisting of the marked forbidden pattern $\mathfrak{p}_l$ followed by $k-1$ fall steps. By applying the previous actions, a path ω' can be written as $\omega' = v\varphi'$, where φ' is the rightmost suffix in ω' beginning from the x-axis and strictly remaining above the x-axis. Clearly, in order to determine the suffix φ' of ω' we ignore the possible points on or below the x-axis which are within the black boxes.

If the suffix φ' does not contain any marked forbidden pattern, then the desired label (0_1) is associated with the path $v(\varphi')^c x$, where $(\varphi')^c$ is the path obtained from φ' by switching rise and fall steps (see Figure 9).

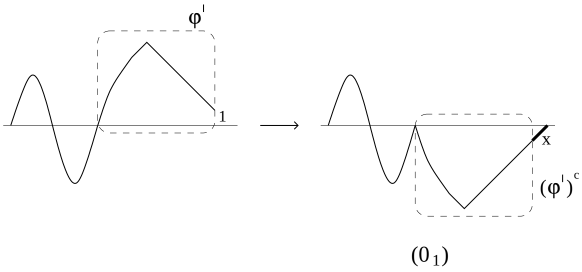

Figure 9: A graphical representation of the actions giving the path labeled (0_1) in case of no marked forbidden patterns in φ'.

If the suffix φ' contains marked forbidden patterns, let z be the leftmost point in φ' having highest ordinate and not strictly contained in a marked forbidden pattern. The desired label (0_1) is associated to the path obtained by concatenating to v a fall step $\overline{x}$ and then the path in φ' running from z to the endpoint of the path and the path running from the initial point in φ' to z. (see Figure 10).

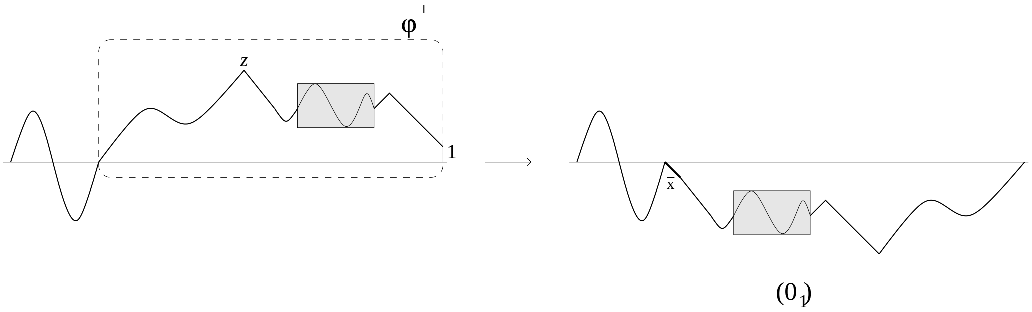

Figure 10: A graphical representation of the cut and paste actions giving the path labeled (0_1) in case of marked forbidden patterns in φ'.

This last mapping can be inverted as follows. Let d be the rightmost fall step in a path ω^* labelled (0_1) such that it begins from the x-axis and each point L_l of the marked forbidden patterns, on its right, has ordinate less than 0. Let $\omega^* = vd\varphi^*$ and H be the rightmost point in φ^* with lowest ordinate. The inverted lattice path of ω^* is given by $v\beta\alpha$, where β is the path in φ^* running from H to the endpoint of the path and α is the path running from the initial point in φ^* to H.

At this point, we have the complete mapping defined by the succession rule (10). For instance, Figure 11 shows the first levels of the generating tree according to the jumping and marked succession rule (10) with the cross-bifix-free set $\mathcal{P} = \{\mathfrak{p}_1(2), \mathfrak{p}_2(3)\} = \{1100, 110100\}$.

The above construction generates 2^C copies of each path having C forbidden patterns such that 2^{C-1} instances are coded by a sequence of labels ending by a marked label, say $(\overline{k})$, and contain an odd number of marked forbidden patterns, and 2^{C-1} instances are coded by a sequence of labels ending by a non-marked label, say (k), and contain an even number of marked forbidden patterns. This is due to the fact that when a path is obtained according to the first production of (10) then no marked forbidden pattern is added. Moreover, when a path is obtained according to the other productions of (10) exactly one marked forbidden pattern is added. In any case, the actions performed to obtain the label (0_1) do not change the number of marked forbidden patterns in the path itself.

The described algorithm generates the set $F^{[\mathcal{P}]}$ according to the number of rise steps. This means that all the paths in F with n rise steps are obtained once. Moreover, for each obtained path ω in $F \backslash F^{[\mathcal{P}]}$, having C forbidden patterns, with n rise steps and (k) as last label of the associated code, a path ω' in $F \backslash F^{[\mathcal{P}]}$ with n rise steps, C forbidden patterns and $(\overline{k})$ as last label of the associated code is also generated having the same form as ω but such that the last forbidden pattern is marked if it is not in ω and vice-versa. This prove the following statement.

Proposition 3.1 *Let $\mathcal{P} = \{\mathfrak{p}_1(j_1), \mathfrak{p}_2(j_2), \ldots, \mathfrak{p}_m(j_m)\}$ be a cross-bifix-free set of patterns, none include in any other, such that $|\mathfrak{p}_l(j_l)|_1 = |\mathfrak{p}_l(j_l)|_0 = j_l$, $1 \leq l \leq m$. The generating tree of the paths in $F^{[\mathcal{P}]}$, according to the number of rise steps, is isomorphic to the tree having its root labelled* (0) *and recursively defined by the succession rule* (10).

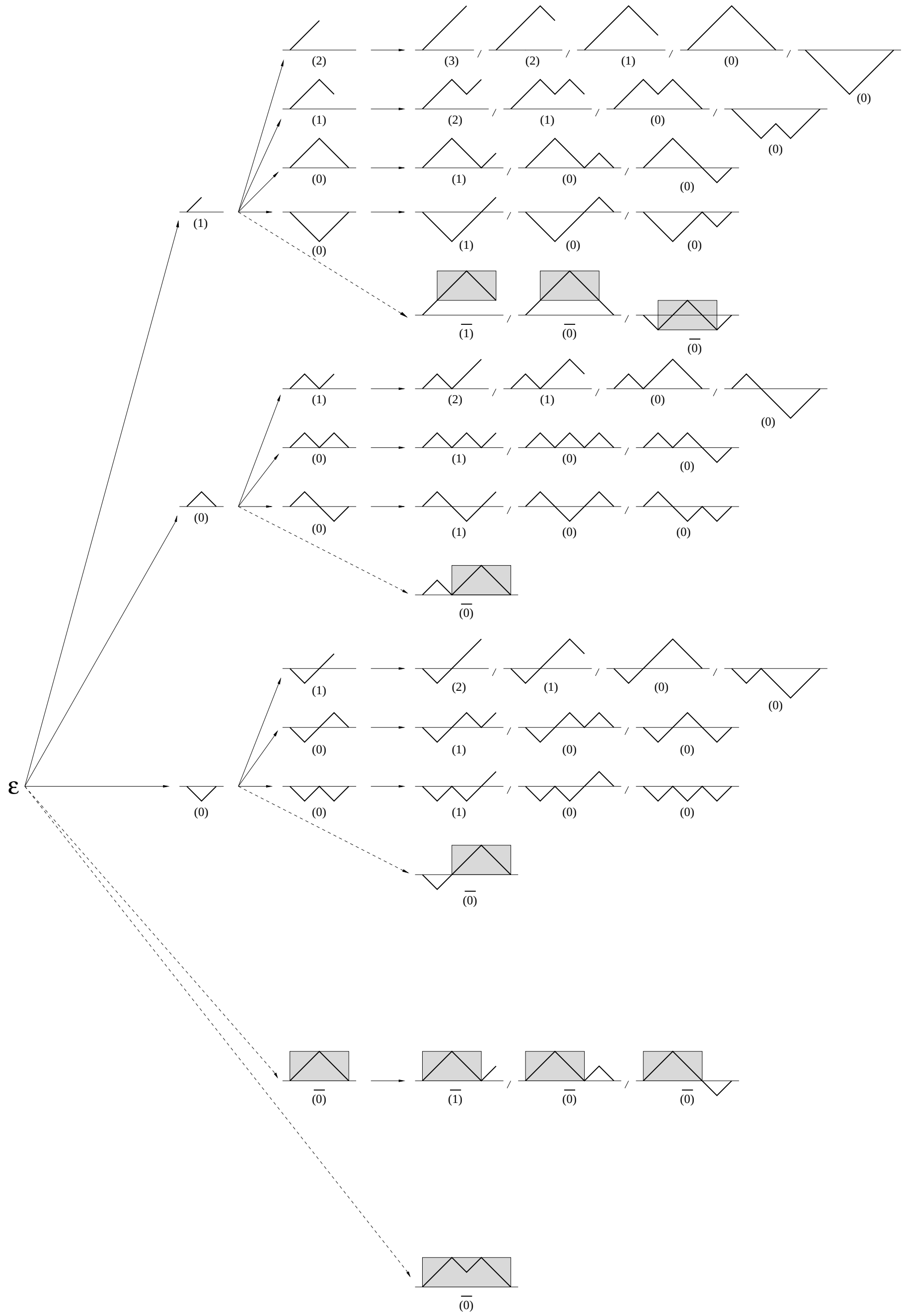

Figure 11: The first levels of the generating tree for $F^{[\mathcal{P}]}$ with $\mathcal{P} = \{1100, 110100\}$.

4 Enumeration of $F^{[\mathcal{P}]}$

In order to obtain the enumeration of the class $F^{[\mathcal{P}]}$ according to the number of rise steps, we use a standard method, called ECO-method, for the enumeration of combinatorial objects which admit recursive descriptions in terms of generating trees, see [2, 3].

Let Z be the set of paths whose instances are coded by a sequence of labels in the generating tree ending by a non-marked zero, S be the set of paths whose instances are coded by a sequence of labels ending by a marked zero, N be the set of paths whose instances are coded by a sequence of labels ending by a non-marked $k \geq 1$ and M be the set of paths whose instances are coded by a sequence of labels ending by a marked $k \geq 1$. Then $F^{[\mathcal{P}]} = (Z \backslash S) \cup (N \backslash M)$.

The succession rule (10) can be written as:

$$\begin{cases} (0) \\ (0) \overset{1}{\rightsquigarrow} (1)(0_2)(0_1) \\ (k) \overset{1}{\rightsquigarrow} (k+1)(k)\cdots(1)(0_2)(0_1) & k \geq 1 \\ (0) \overset{j_1}{\rightsquigarrow} (\overline{0_2}) \\ (k) \overset{j_1}{\rightsquigarrow} (\overline{k})(\overline{k-1})\cdots(\overline{1})(\overline{0_2})(\overline{0_1}) & k \geq 1 \\ \vdots \\ (0) \overset{j_m}{\rightsquigarrow} (\overline{0_2}) \\ (k) \overset{j_m}{\rightsquigarrow} (\overline{k})(\overline{k-1})\cdots(\overline{1})(\overline{0_2})(\overline{0_1}) & k \geq 1. \end{cases} \tag{11}$$

Let us denote by $n(\omega)$ the number of rise steps of a path $\omega \in F$ and by $h(\omega)$ the last point's ordinate of ω itself. From the succession rule (11) we have:

$$\begin{aligned} Z(x,1) &= 1 + 2xZ(x,1) + 2xN(x,1) + (x^{j_1} + \cdots + x^{j_m})S(x,1) + 2(x^{j_1} + \cdots + x^{j_m})M(x,1), \\ S(x,1) &= 2xS(x,1) + 2xM(x,1) + (x^{j_1} + \cdots + x^{j_m})Z(x,1) + 2(x^{j_1} + \cdots + x^{j_m})N(x,1), \\ N(x,y) &= xyZ(x,1) + \sum_{\omega \in N} \sum_{i=1}^{h(\omega)+1} x^{n(\omega)+1}y^i + \sum_{\omega \in M} \sum_{i=1}^{h(\omega)} x^{n(\omega)+j_1}y^i + \cdots + \sum_{\omega \in M} \sum_{i=1}^{h(\omega)} x^{n(\omega)+j_m}y^i, \\ M(x,y) &= xyS(x,1) + \sum_{\omega \in M} \sum_{i=1}^{h(\omega)+1} x^{n(\omega)+1}y^i + \sum_{\omega \in N} \sum_{i=1}^{h(\omega)} x^{n(\omega)+j_1}y^i + \cdots + \sum_{\omega \in N} \sum_{i=1}^{h(\omega)} x^{n(\omega)+j_m}y^i. \end{aligned}$$

Since $\sum_{\omega \in N} \sum_{i=1}^{h(\omega)+1} x^{n(\omega)+1}y^i = \sum_{\omega \in N} x^{n(\omega)+1} \left(\frac{y^{h(\omega)+2}-y}{y-1}\right) = \frac{xy^2}{y-1}N(x,y) - \frac{xy}{y-1}N(x,1)$ going on in the same way with the other terms, then we obtain:

$$\begin{aligned} N(x,y) &= xyZ(x,1) + \frac{xy^2}{y-1}N(x,y) - \frac{xy}{y-1}N(x,1) + \frac{(x^{j_1}+\ldots+x^{j_m})y}{y-1}M(x,y) - \\ &\quad -\frac{(x^{j_1}+\ldots+x^{j_m})y}{y-1}M(x,1), \\ M(x,y) &= xyS(x,1) + \frac{xy^2}{y-1}M(x,y) - \frac{xy}{y-1}M(x,1) + \frac{(x^{j_1}+\ldots+x^{j_m})y}{y-1}N(x,y) - \\ &\quad -\frac{(x^{j_1}+\ldots+x^{j_m})y}{y-1}N(x,1). \end{aligned}$$

Since $T(x,y) = N(x,y) - M(x,y)$ then:

$$\begin{aligned} T(x,y) &= xy(Z(x,1)-S(x,1)) + \frac{(xy^2-(x^{j_1}+\ldots+x^{j_m})y)}{y-1}T(x,y) - \\ &\quad -\frac{(xy-(x^{j_1}+\ldots+x^{j_m})y)}{y-1}T(x,1), \end{aligned}$$

that is,

$$T(x,y)(y-1-xy^2+(x^{j_1}+\ldots+x^{j_m})y) = xy(y-1)(Z(x,1)-S(x,1)) - (xy-(x^{j_1}+\ldots+x^{j_m})y)T(x,1).$$

Let

$$y_0(x) = \frac{1+x^{j_1}+\ldots+x^{j_m}-\sqrt{(x^{j_1}+\ldots+x^{j_m}+1)^2-4x}}{2x}$$

be a solution of $xy^2-(x^{j_1}+\ldots+x^{j_m}+1)y+1=0$. Then we have the desired equation according to the number of ones, only:

$$T(x,1) = \frac{y_0(x)-1}{1-x^{j_1-1}+\ldots+x^{j_m-1}}(Z(x,1)-S(x,1)).$$

Since

$$W(x,1) = Z(x,1)-S(x,1) = \frac{1+2x(1-x^{j_1-1}+\ldots+x^{j_m-1})T(x,1)}{1-2x+x^{j_1}+\ldots+x^{j_m}},$$

then we have:

$$T(x,1) = \frac{y_0(x)-1}{(1-x^{j_1-1}-\cdots-x^{j_m-1})(1+x^{j_1}+\cdots+x^{j_m}-2xy_0(x))},$$

$$W(x,1) = \frac{1}{(1+x^{j_1}+\cdots+x^{j_m}-2xy_0(x))}.$$

Therefore the generating function $F_{j_1,\ldots,j_m}(x) = T(x,1)+W(x,1)$ for the words $\omega \in F^{[\mathcal{P}]}$ according to the number of ones is:

$$F_{j_1,\cdots,j_m}(x) = \frac{y_0(x)-x^{j_1-1}-\cdots-x^{j_m-1}}{(1-x^{j_1-1}-\cdots-x^{j_m-1})(1+x^{j_1}+\cdots+x^{j_m}-2xy_0(x))}.$$

Let us remark that the generating function $F_{j_1,\ldots,j_m}(x)$ depends only on the number of ones in each forbidden pattern, so it enumerates all the sets of binary words in F avoiding cross-bifix-free forbidden patterns with $j_1,\ldots,j_m$ ones (or zeroes) independently from their shapes.

For instance, considering any set $\mathcal{P}=\{\mathfrak{p}_1(2),\mathfrak{p}_2(3)\}$, which includes also $\mathcal{P}=\{1100,110100\}$, the first numbers of the sequence enumerating the binary words in $F^{[\mathcal{P}]}$, according to the number of ones, are:

$$1,3,9,26,74,210,598,1713,4939,14330,41814,\ldots$$

being

$$F_{2,3}(x) \quad = \quad \frac{\frac{1+x^2+x^3-\sqrt{(x^2+x^3+1)^2-4x}}{2x}-x-x^2}{(1-x-x^2)\sqrt{(x^2+x^3+1)^2-4x}},$$

the associated generating function.

Note that, in the simple case $j=1$, we have either $\mathcal{P}=\mathfrak{p}(1)=\{10\}$ or $\mathcal{P}=\mathfrak{p}(1)=\{01\}$ exclusively. Adapting to the case $j=1$ the above equations $Z(x,1),S(x,1),N(x,y)$ and $M(x,y)$ and applying the previous computation, we obtain $W(x,1)=1+xW(x,1)$ and $T(x,y)=xyW(x,1)+xyT(x,y)$, that is

$$W(x,1)=\frac{1}{1-x}, \qquad T(x,1)=\frac{x}{(1-x)^2}.$$

Therefore the generating function $F_1(x)$ is:

$$F_1(x)=T(x,1)+W(x,1)=\frac{1}{(1-x)^2},$$

that is, $F_1(x)$ is the generating function of the succession $x_n=n+1$, $n\geq 1$. Indeed, when $j=1$, the set $F^{[\mathcal{P}]}$ of binary words with n ones and avoiding the forbidden pattern $\mathcal{P}=\mathfrak{p}(1)=\{10\}$ (resp. $\mathcal{P}=\mathfrak{p}(1)=\{01\}$) is $F^{[\mathcal{P}]}=\{0^m1^n\,|\,0\leq m\leq n\}$ (resp. $F^{[\mathcal{P}]}=\{1^n0^m\,|\,0\leq m\leq n\}$).

References

[1] A. Apostolico and M. Atallah, *Compact recognizers of episode sequences*, Inform. Comput., 174 (2002) 180–192.

[2] C. Banderier, M. Bousquet-Mélou, A. Denise, P. Flajolet, D. Gardy and D. Gouyou-Beauchamps, *Generating functions for generating trees*, Discrete Math., 246 (2002) 29–55.

[3] E. Barcucci, A. Del Lungo, E. Pergola and R. Pinzani, *ECO: a methodology for the Enumeration of Combinatorial Objects*, J. Differ. Equ. Appl., 5 (1999) 435–490.

[4] S. Bilotta, E. Grazzini and E. Pergola, *Primitive Dyck path as an invariant class for the enumeration*, In Proc. GASCom 2012.

[5] S. Bilotta, E. Pergola and R. Pinzani, *A new approach to cross-bifix-free sets*, IEEE Trans. Inf. Theory, 58 (2012) 4058–4063.

[6] S. Bilotta, E. Grazzini, E. Pergola and R. Pinzani, *Avoiding cross-bifix-free binary words*, Acta Inform., 50 (2013) 157–173.

[7] Y. M. Chee, H. M. Kiah, P. Purkayastha and C. Wang, *Cross-bifix-free codes within a constant factor of optimality*, IEEE Trans. Inform. Theory, 59 (2013) 4668–4674.

[8] L. Comtet, *Advanced Combinatorics: The Art of Finite and Infinite Expansions*, D. Reidel Publishing Company, 1974.

[9] S. Corteel, Séries génératrices exponentielles pour les ECO-systèmes signés, In proc. FPSAC 2000.

[10] L. Ferrari, E. Pergola, R. Pinzani and S. Rinaldi, *Jumping succession rules and their generating functions*, Discrete Math., 271 (2003) 29–50.

[11] P. Flajolet, W. Szpankowski and B. Valle, *Hidden word statistics*, J. ACM, 53 (2006) 147–183.

[12] L. J. Guibas and M. Odlyzko, *String overlaps, pattern matching, and nontransitive games*, J. Combin. Theory Ser. A, 30 (1981) 183–208.

[13] K. H. Kim, M. S. Putcha and F. W. Roush, *Some combinatorial properties of free semigroups*, J. London Math. Soc., 16 (1977) 397–402.

[14] S. Kumar and E. H. Spafford, *A pattern matching model for misuse intrusion detection*, In Proc. NCSC 1994, 11–21.

[15] P. T. Nielsen, *A Note on Bifix-Free Sequences*, IEEE Trans. Inform. Theory, 29 (1973) 704–706.

[16] I. Rigoutsos, A. Floratos, L. Parida, Y. Gao and D. Platt, *The emergence of pattern discovery techniques in computational biology*, Metabolic Engineering, 2 (2000) 159–177.

[17] R. Sedgewick and P. Flajolet, *An Introduction to the Analysis of Algorithms*, Chapman-Hall, 1995.

[18] M. Waterman, *Introduction to Computational Biology*, Addison-Wesley, 1995.