On the generation and enumeration of some classes of convex polyominoes

A. Del Lungo*

E. Duchi

École des Hautes Études en Sciences Sociales
54 Boulevard Raspail, 75006 Paris, France
duchi@ehess.fr

A. Frosini and S. Rinaldi

Dipartimento di Scienze Matematiche e Informatiche
Pian dei Mantellini, 44, Siena, Italy
{frosini, rinaldi}@unisi.it

Submitted: Jul 29, 2003; Accepted: Jul 5, 2004; Published: Sep 13, 2004
Mathematics Subject Classifications: 05A15

Abstract

ECO is a method for the recursive generation, and thereby also the enumeration of classes of combinatorial objects. It has already found successful application in recent literature both to the exhaustive generation and to the uniform random generation of various objects classified according to several parameters of interest, as well as to their enumeration.

In this paper we extend this approach to the generation and enumeration of some classes of convex polyominoes. We begin with a review of the ECO method and of the closely related notion of a *succession rule*.

From this background, we develop the following principal findings:

i) ECO constructions for both column-convex and convex polyominoes;

ii) translations of these constructions into succession rules;

iii) the consequent deduction of the generating functions of column-convex and of convex polyominoes according to their semi-perimeter, first of all analytically by means of the so-called *kernel method*, and then in a more novel manner by drawing on some ideas of Fedou and Garcia;

iv) algorithms for the exhaustive generation of column convex and of convex polyominoes which are based on the ECO constructions of these object and which are shown to run in constant amortized time.

*died 1st June, 2003

1 Introduction

A *polyomino* is a finite union of cells of the square lattice $Z \times Z$ with simply connected interior. In the half century since Solomon Golomb used the term in his seminal article [22], the study of polyominoes has proved a fertile topic of research. By this period in the mid-1950s, it was clearly a timely notion in discrete models, as the increasingly influential work of Neville Temperley, on problems drawn from statistical mechanics and molecular dynamics [29], and of John Hammersely, dealing with percolation [23], bear witness. More recent years have seen the treatment of numerous related problems, such as the problem of covering a polyomino by rectangles [9] or problems of tiling regions by polyominoes [5, 12]. But, at the same time, there remain many challenging, open problems, starting with general enumeration problem. Here, while the number a_n of polyominoes with n cells has been determined only for small n (up to $n = 94$ in [26]), it is known that *asymptotically* there is geometrical growth:

$$lim_n \{a_n\}^{1/n} = \mu, \quad 3.72 < \mu < 4.64.$$

Consequently, in order to probe further, several subclasses of polyominoes have been introduced on which to hone enumeration techniques. One very natural subclass is that of *convex* polyominoes with which we are concerned in this paper. A polyomino is said to be *column-convex* [*row-convex*] when its intersection with any vertical [horizontal] line of cells in the square lattice is connected (see Fig. 1 (a)), and *convex* when it is both column and row-convex (see Fig. 1 (b)). The *area* of a polyomino is just the number of cells it contains, while its *semi-perimeter* is half the number of edges of cells in going around the boundary. Thus, a convex polyomino has the agreeable property that its semi-perimeter is the sum of the numbers of its rows and columns. Moreover, any convex polyomino is contained in a rectangle in the square lattice which has the same semi-perimeter.

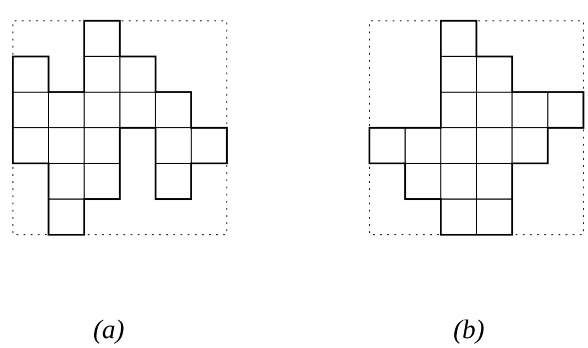

Figure 1: (a) a column-convex (but not convex) polyomino; (b) a convex polyomino.

In fact, the number f_n of convex polyominoes with semi-perimeter $n+2$ was determined by Delest and Viennot, in [14]:

$$f_{n+2} = (2n+11)4^n - 4(2n+1)\binom{2n}{n}, \quad n \geq 0; \quad f_0 = 1, \quad f_1 = 2. \tag{1}$$

This is an instance of sequence $A005436$ in [28], the first few terms being:

$$1, 2, 7, 28, 120, 528, 2344, 10416, \ldots.$$

Thus, for example, there are seven convex polyominoes with semi-perimeter 4, as shown in Figure 2.

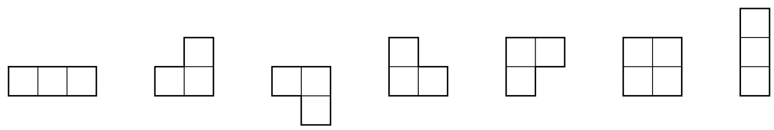

Figure 2: The seven convex polyominoes with semi-perimeter 4.

In [14], the enumeration of convex polyominoes is *via* encoding in context-free languages generated by non-ambiguous languages (the so-called DSV methodology) coupled with generating functions in two steps:

1. each polyomino is classified into one of three types and then encoded accordingly as a word in a corresponding language; and

2. for each of the three languages used, the grammar productions are translated into an algebraic system of equations from which the generating function can be deduced.

The upshot is to show that the generating function $f(x)$ for convex polyominoes enumerated by semi-perimeter is given by:

$$f(x) = x^2 \sum_{n \geq 0} f_n x^n = x^2 \left(\frac{1 - 6x + 11x^2 - 4x^3 - 4x^2\sqrt{1-4x}}{(1-4x)^2} \right), \tag{2}$$

from which (1) follows.

In the wake of this pioneering effort, (2) has been re-derived by other analytical means in [10, 24] and, much more recently, through a bijective proof in [7]. Indeed, in their paper [10], Lin and Chang do rather more, refining the enumeration to give the generating function for the number of convex polyominoes with $k+1$ columns and $j+1$ rows (and so semi-perimeter $k+j+2$), where $k, j \geq 0$. From this result, in turn, Gessel was able to infer, in a brief note [21], that the number of such polyominoes is

$$\frac{k+j+kj}{k+j}\binom{2k+2j}{2k} - 2(k+j)\binom{k+j-1}{k}\binom{k+j-1}{j}. \tag{3}$$

The enumeration of column-convex polyominoes that form another part of our present subject matter has an even older history, going back to Temperley's paper [29] in 1956. However, the determination of the generating function $g(x)$ for column-convex polyominoes indexed by semi-perimeter was obtained only in the late 1980s by Delest in [13], as a further application of encoding in context-free languages, together with appeal to the computer algebra program MACSYMA. The resulting expression for $g(x)$ is rather more complicated than that for $f(x)$ in (2):

$$(1-x)\left(1-\frac{2\sqrt{2}}{\left(3\sqrt{2}-\sqrt{1+x+\sqrt{\frac{(x^2-6x+1)(1+x)^2}{(1-x)^2}}}\right)}\right). \tag{4}$$

The number g_n of column-convex polyominoes with semi-perimeter $n+2$ is then the coefficient of x^n in $g(x)$; these coefficients are an instance of sequence $A005435$ in [28], the first few terms being

$$1, 2, 7, 28, 122, 558, 2641, 12822, \ldots.$$

As with (2), Lin and Chang also provided a generalization of (4) in their paper [10]. Perhaps because there does not appear to be any closed expression for the coefficients $g_(n)$, (4) continues to exercise interest, an alternative proof being given in [20] by Feretic. Brak, Enting, and Gutman [8] had shown earlier how to obtain $g(x)$ using Temperley's methodology and *Mathematica*, giving, in default of a closed form for g_n, the following asymptotic expansion:

$$g_{n-2} \sim (3+2\sqrt{2})^{n-1/2}\, n^{-3/2}\left[c_0+\frac{c_1}{n}+O\left(\frac{1}{n^2}\right)\right], \tag{5}$$

where $c_0 = 0,102834615\ldots$, $c_1 = 0,038343814\ldots$.

In the course of our analysis, it is helpful to be able to call upon results for two further familiar classes of polyominoes defined by means of *proper lattice paths* in the square lattice, namely the *directed-convex polyominoes* and the *parallelogram polyominoes*. By a proper lattice path between two lattice points in the square lattice is meant a path made up in some combination of unit steps up or to the right along the lines of the lattice. A polyomino is said to be *directed* when each of its lattice points can be reached from its bottom left-hand corner by a proper lattice path. In turn, a polyomino is *directed-convex* if it is both directed and convex (so, for example, the polyomino in Fig. 3 (a) is directed-convex, whereas the convex polyomino already illustrated in Fig. 1 (b) fails to be directed). The number of directed-convex polyominoes with semi-perimeter $n+2$ is the central binomial coefficient

$$\binom{2n}{n}, \tag{6}$$

giving an instance of sequence $A000984$ in [28].

A significant subclass of the directed-convex polyominoes that has attracted much attention arises by requiring that the boundary of a polyomino consists of two proper lattice paths between two lattice points that otherwise do not intersect. A polyomino with such a boundary is called a *parallelogram polyomino*, the two proper lattice paths defining its boundary being known as the *upper* and *lower* paths, in the obvious sense that, except at the end-points, one path runs *above* the other (thus, the polyomino in Fig. 3 (b) is a parallelogram polyomino, but that in Fig. 3 (a) is not). In this case, it is well-known that the number of parallelogram polyominoes with semi-perimeter $n+2$ is the $(n+1)$-st *Catalan* number C_{n+1}, where

$$C_n = \frac{1}{n+1}\binom{2n}{n}; \tag{7}$$

the sequence of Catalan numbers is $A000108$ in [28]. A further appealing fact is that the total number of cells in the parallelogram polyominoes with semi-perimeter $n+2$ is 4^n, giving rise to the celebrated problem of the *Catalan jigsaw* highlighted in [32].

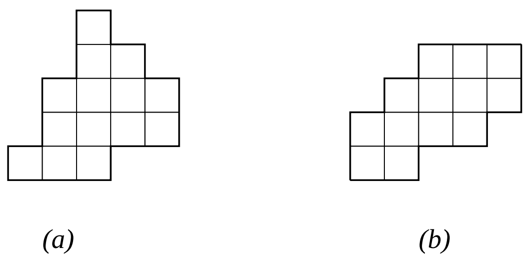

Figure 3: (a) a directed-convex polyomino; (b) a parallelogram polyomino.

Our aim here is to provide a *unifying* approach to the generation and enumeration of certain classes of polyominoes, applicable, for instance, to the directed-convex, the convex, and the column-convex polyominoes. In this enterprize, we draw for our inspiration on [6] which proposes

> *a single method to get, for any "natural" class of column convex polyominoes, a functional equation that implicitly defines its generating function,*

where, indeed, the generating function takes account of several parameters including area and semi-perimeter. But our technique of choice is the *ECO method*, for a survey of which we refer to [4]. The *crux* of this method is the recursive generation of classes of combinatorial objects through *local expansions* of the objects of one size that yield every object of the next size once and only once. If this recursive procedure has sufficient regularity, it can be translated into a formal system known as a *succession rule*. While the principles at work here had been employed earlier informally, the definition of a succession rule seems to have been first formalized in [30, 31]. It was then found to be an apt tool for the ECO method. A closely associated notion is that of a *generating tree*, which provides a handy means of representing succession rules, and this perspective was explored in [3]. The focus in [3] is on how the form of a succession rule is linked to the resulting generating function, leading to a classification of succession rules as *rational, algebraic,* or *transcendental* according to the type of generating function that arises. The computation of these generating functions is obtained by using the *kernel method*. We supply an introductory summary of the ECO method, including succession rules and generating trees, in the next section. (Since the ECO method is an attempt to capture a natural and attractive approach to the generation of combinatorial objects, it is no surprise that similar algorithms have been formulated independently, and, indeed, at about the same time, for example, under the names *reverse search* in [1], and *canonical construction path* in [25].)

The first part of the paper is devoted to proving (2) by breaking the task into the following stages that together make for a pleasingly simple result:

1. an ECO construction is developed for the set of convex polyominoes (Section 2);
2. the associated succession rule is then deduced from this construction; its generating function is just $f(x)/x^2$, the generating function of the sequence $\{f_n\}_{n\geq 0}$ (Section 2.1);
3. the generating function of the succession rule is computed by standard analytical methods as given in [3], especially the kernel method, which involves solving a system of functional equations (Section 3.1);
4. this result is re-derived in novel fashion, starting from a method proposed by Fédou and Garcia, in [17], for some algebraic succession rules, and extending it to the present case on noting that a convex polyomino with semi-perimeter $n+2$ has a representation as a word of length n of a non-commutative formal power series over an infinite alphabet; this non-commutative power series admits a decomposition in terms of some auxiliary power series which yields an algebraic system of equations on taking commutative images; and the solution of this system is then the generating function $f(x)$ as in (2) (Section 3.2).

It is worth noting here that the approach summarized in Step 4 has wider applicability in the solution of functional equations arising from the ECO method where the kernel method may fail.

Similarly, we also derive an ECO construction for column-convex polyominoes, deduce the associated succession rule, and then apply the methodologies described in Steps 3 and 4 to pass from the succession rule to a system of equations satisfied by the generating function $g(x)$ (Section 3.3). This system can be solved using MAPLE to give $g(x)$ as in (4).

In latter part of the paper we examine the *exhaustive generation* of the classes of convex and column-convex polyominoes. The aim in studying the exhaustive generation of combinatorial objects is to describe *efficient algorithms* to list all the objects. Algorithms of this sort find application in many areas: hardware and software testing, combinatorial chemistry and the design of pharmaceutical compounds, coding theory and reliability theory, and computational biology, to name a few. Moreover, these algorithms can yield supplementary information about the objects under review.

The primary measure of performance for the efficiency of an algorithm for generating combinatorial objects is that the amount of computational time taken should remain proportional to the number of objects to be generated. Thus, an algorithm for exhaustive generation is regarded as *efficient* when it requires only a constant amount of computation *per* object, in an amortized sense, algorithms attaining this benchmark being said to have the *Constant Amortized Time* or *CAT* property.

In [2], it is shown that an ECO construction leads to an algorithm for the generation of the objects being constructed. In Section 4, we use the ECO construction defined in Section 2 coupled with the strategy proposed in [2], to describe two algorithms, one for generating convex polyominoes and the other for column-convex polyominoes. We then confirm that both have the CAT property.

2 An ECO operator for the class of convex polyominoes

ECO (Enumerating Combinatorial Objects) is a method for the enumeration and the recursive construction of a class of combinatorial objects, $\mathcal{O}$, by means of an operator ϑ which performs "local expansions" on the objects of $\mathcal{O}$. More precisely, let p be a parameter on $\mathcal{O}$, such that $|\mathcal{O}_n| = |\{O \in \mathcal{O} : p(O) = n\}|$ is finite. An operator ϑ on the class $\mathcal{O}$ is a function from $\mathcal{O}_n$ to $2^{\mathcal{O}_{n+1}}$, where $2^{\mathcal{O}_{n+1}}$ is the power set of $\mathcal{O}_{n+1}$.

Proposition 2.1 Let ϑ be an operator on $\mathcal{O}$. If ϑ satisfies the following conditions:

1. for each $O' \in \mathcal{O}_{n+1}$, there exists $O \in \mathcal{O}_n$ such that $O' \in \vartheta(O)$,

2. for each $O, O' \in \mathcal{O}_n$ such that $O \neq O'$, then $\vartheta(O) \cap \vartheta(O') = \emptyset$,

then $\mathcal{F}_{n+1} = \{\vartheta(O) : O \in \mathcal{O}_n\}$ is a partition of $\mathcal{O}_{n+1}$.

ECO method was successfully applied to the enumeration of various classes of walks, permutations, and polyominoes. We refer to [4] for further details, and examples.

The recursive construction determined by ϑ can be suitably described through a *generating tree*, i.e. a rooted tree whose vertices are objects of $\mathcal{O}$. The root of the tree is the object of $\mathcal{O}$ having the minimum size (possibly the empty object). The objects having the same value of the parameter p lie at the same level, and the sons of an object are the objects it produces through ϑ.

In this section we define an ECO operator for the recursive construction of the set of convex polyominoes. First, we partition the set of convex polyominoes $\mathcal{C}$ into four disjoint subsets, denoted by $\mathcal{C}_b$, $\mathcal{C}_a$, $\mathcal{C}_r$, and $\mathcal{C}_g$. In order to define the four classes, let us consider the following conditions on convex polyominoes:

U1 : the uppermost cell of the rightmost column of the polyomino has the maximal ordinate among all the cells of the polyomino;

U2 : the lowest cell of the rightmost column of the polyomino has the minimal ordinate among all the cells of the polyomino.

(b) (a)

Figure 4: A convex polyomino in $\mathcal{C}_b$, on the left, and one polyomino in $\mathcal{C}_a$, on the right.

We are now able to set the following definitions:

i) $\mathcal{C}_b$ is the set of convex polyominoes having at least two columns, satisfying conditions **U1** and **U2**, and such that the uppermost cell of the rightmost column has the same ordinate as the uppermost cell of the column on its left (Fig. 4 (b)).

(r) (g)

Figure 5: A convex polyomino in $\mathcal{C}_r$, on the left, satisfying **U1** but not **U2** and one polyomino in $\mathcal{C}_g$, on the right.

ii) $\mathcal{C}_a$ is the set of convex polyominoes not in $\mathcal{C}_b$, and satisfying conditions **U1** and **U2** (see Fig. 4 (a)).

Let us remark that, according to such definition, all convex polyominoes made only of one column lie in the class $\mathcal{C}_a$.

iii) $\mathcal{C}_r$ is the set of convex polyominoes that satisfy only one of the conditions **U1** and **U2** (for example Figure 5 (r), depicts a polyomino that satisfies condition **U1** but not **U2**).

iv) $\mathcal{C}_g$ is the set of remaining convex polyominoes, i.e. those satisfying neither **U1** nor **U2** (see Fig. 5 (g)).

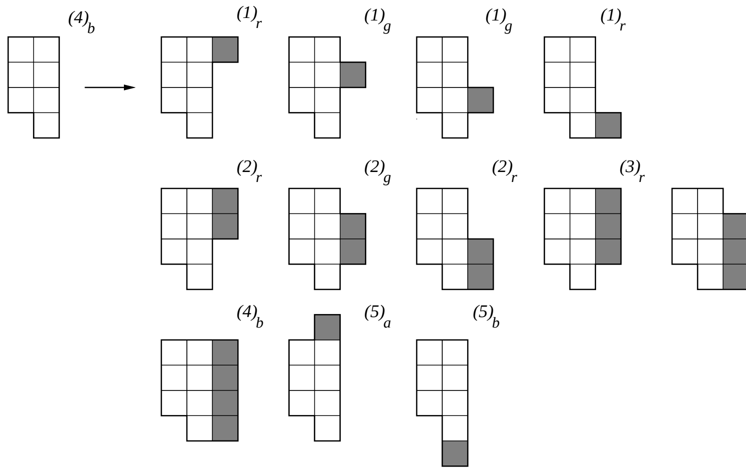

Figure 6: The ECO operator applied to a polyomino in the class $\mathcal{C}_b$.

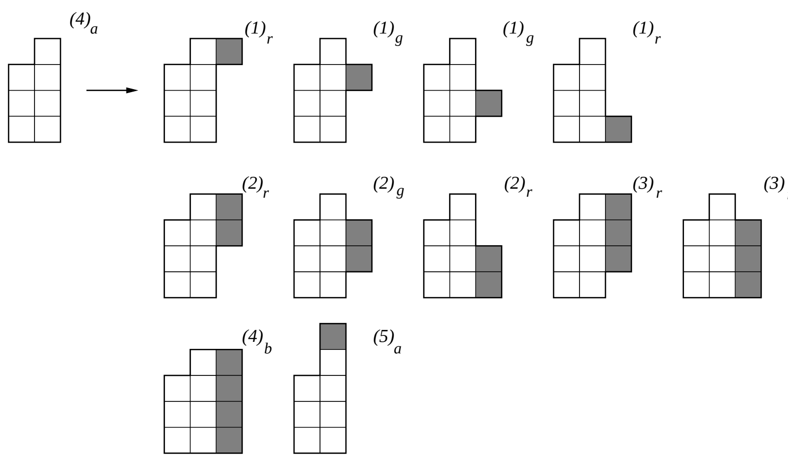

Figure 7: The ECO operator applied to a polyomino in the class $\mathcal{C}_a$.

The ECO operator we are going to define, namely ϑ, performs local expansions on the rightmost column of any polyomino of semi-perimeter $n+2$, thus producing a set of polyominoes of semi-perimeter $n+3$. More precisely, the operator ϑ performs the following set of expansions on any convex polyomino P, with semi-perimeter $n+2$ and k cells in the rightmost column:

- for any $i = 1, \dots, k$ the operator ϑ glues a column of length i to the rightmost column of P; this can be done in $k - i + 1$ possible ways.

The previous operation produces $k + (k-1) + \ldots + 2 + 1$ polyominoes with semi-perimeter $n+3$. Moreover, the operator performs some other transformations on convex polyominoes of classes $\mathcal{C}_b$, $\mathcal{C}_a$, and $\mathcal{C}_r$, according to the belonging class:

- if $P \in \mathcal{C}_b$, then the operator ϑ produces two more polyominoes, one by gluing a cell onto the top of the rightmost column of P, and another by gluing a cell onto the bottom

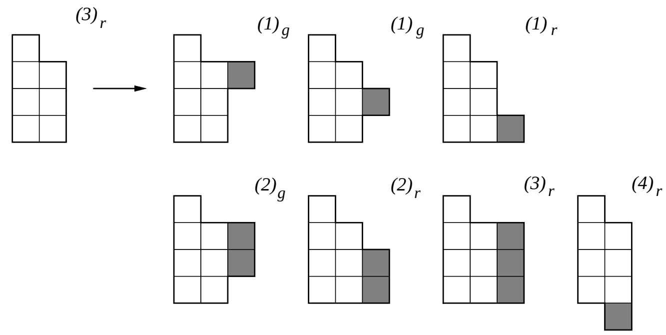

Figure 8: The ECO operator applied to a polyomino in the class $\mathcal{C}_r$.

of the rightmost column of P (Figure 6 depicts the whole set of the expansions performed by ϑ on a polyomino of the class $\mathcal{C}_b$).

- if $P \in \mathcal{C}_a$, then the operator ϑ produces one more polyomino by gluing a cell onto the top of the rightmost column of P (Fig. 7).

- if $P \in \mathcal{C}_r$, we have two cases:

  if P satisfies the condition **U1**, then the operator ϑ glues a cell onto the top of the rightmost column of P;

  else, the operator ϑ glues a cell on the bottom of the rightmost column of P (Fig. 8).

The ECO operator applied to polyominoes in $\mathcal{C}_g$ makes no addictive expansions, as it is graphically explained in Fig. 9.

The reader can easily check that the operator ϑ produces satisfies conditions 1. and 2. of Proposition 2.1.

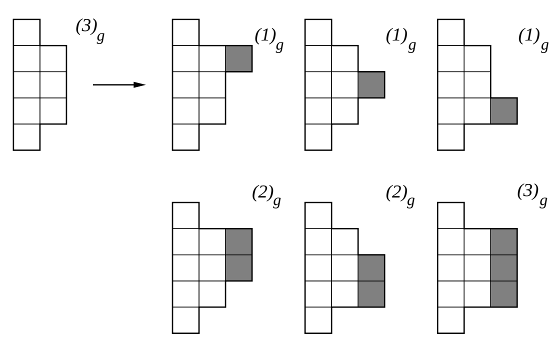

Figure 9: The ECO operator applied to a polyomino in the class $\mathcal{C}_g$.

2.1 The succession rule associated with ϑ

The next step consists in translating the previous construction into a set of equations whose solution is the generating function for convex polyominoes. To achieve this purpose, we must introduce the second ingredient of our work, the concept of *succession rule*. Before turning to a formal definition, we present an illustrative example.

A polyomino in $\mathcal{C}_i$, $i \in \{a, b, g, r\}$ with k cells in the rightmost column can be represented by a label $(k)_i$. Let us take as an example, the polyomino in Fig. 8, with label $(3)_r$; according to the figure, the performance of the ECO operator on the polyomino can be sketched by the production:

$$(3)_r \rightsquigarrow (1)_g\ (1)_g\ (1)_r\ (2)_g\ (2)_r\ (3)_r\ (4)_r,$$

meaning that the polyomino produces through ϑ two polyominoes with label $(1)_g$, and polyominoes with labels $(1)_r$, $(2)_g$, $(2)_r$, $(3)_r$, $(4)_r$.

More generally, the performance of the ECO operator on a generic polyomino can be sketched by the following set of *productions*:

$$\begin{cases} (k)_g \rightsquigarrow \prod_{j=1}^{k} (j)_g^{k-j+1} \\ \\ (k)_r \rightsquigarrow \prod_{j=1}^{k-1} (j)_g^{k-j}\ \prod_{j=1}^{k+1} (j)_r \\ \\ (k)_a \rightsquigarrow \prod_{j=1}^{k-2} (j)_g^{k-j-1}\ \prod_{j=1}^{k-1} (j)_r^2\ (k)_b\ (k+1)_a \\ \\ (k)_b \rightsquigarrow \prod_{j=1}^{k-2} (j)_g^{k-j-1}\ \prod_{j=1}^{k-1} (j)_r^2\ (k)_b\ (k+1)_a\ (k+1)_b, \end{cases} \tag{8}$$

where k can assume all positive integer values, and the power notation is used to express repetitions, that is $(i)^j$ is short for the repetition of (i) j times.

The system constituted by:

1. the label $(1)_a$ (often called the *axiom* of the rule); it is the label of the polyomino with semi-perimeter 2;

2. the sets of productions defined in (8),

forms a *succession rule*, which we call Ω. As an example, for $k = 1, 2, 3$ we have the following productions of Ω:

$$\begin{array}{ll}
(1)_g \rightsquigarrow (1)_g & (1)_r \rightsquigarrow (1)_r\ (2)_r \\
(2)_g \rightsquigarrow (1)_g\ (1)_g\ (2)_g & (2)_r \rightsquigarrow (1)_g\ (1)_r\ (2)_r\ (3)_r \\
(3)_g \rightsquigarrow (1)_g\ (1)_g\ (1)_g\ (2)_g\ (2)_g\ (3)_g & (3)_r \rightsquigarrow (1)_g\ (1)_g\ (1)_r\ (2)_g\ (2)_r\ (3)_r\ (4)_r \\
 & \\
(1)_a \rightsquigarrow (1)_b\ (2)_a & (1)_b \rightsquigarrow (1)_b\ (2)_a\ (2)_b \\
(2)_a \rightsquigarrow (1)_r\ (1)_r\ (2)_b\ (3)_a & (2)_b \rightsquigarrow (1)_r\ (1)_r\ (2)_b\ (3)_a\ (3)_b \\
(3)_a \rightsquigarrow (1)_g\ (1)_r\ (1)_r\ (2)_r\ (2)_r\ (3)_b\ (4)_a & (3)_b \rightsquigarrow (1)_g\ (1)_r\ (1)_r\ (2)_r\ (2)_r\ (3)_b\ (4)_a\ (4)_b.
\end{array}$$

The rule Ω can be graphically represented by means of a *generating tree*, i.e. a rooted tree whose vertices are labelled with the labels of the rule. In practice:

1) the root is labelled with the axiom $(1)_a$;

2) each node with label $(k)_t$ produces a set of *sons* whose labels are determined by the production of $(k)_t$ in the rule.

Figure 10, (a), depicts the first levels of the generating tree of the ECO operator ϑ, while Figure 10, (b), shows the first levels of the generating tree of Ω. The two generating trees are naturally isomorphic, and then throughout the paper we will treat them as the same generating tree.

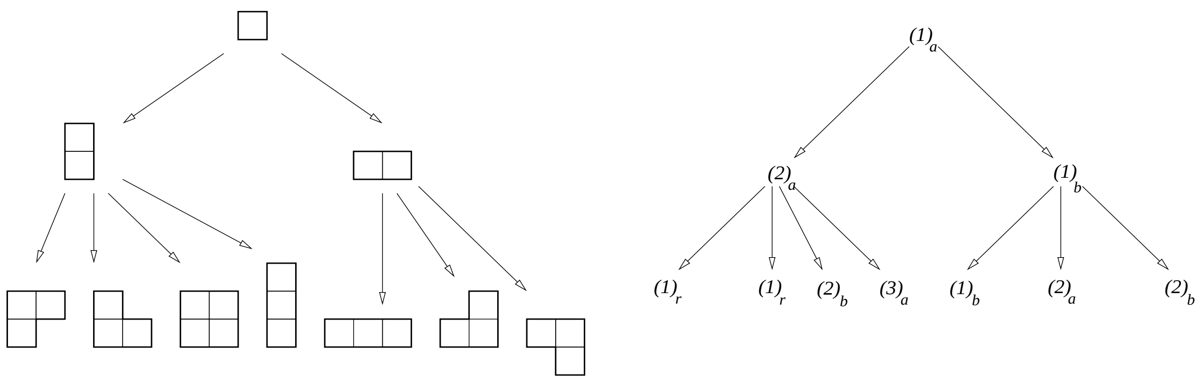

Figure 10: The first levels of the two isomorphic generating trees: of the ECO operator ϑ, on the left, and of the succession rule Ω, on the right.

Remark 1. Each system of the form

$$\begin{cases} (a) \\ (k) \rightsquigarrow (e_1(k))(e_2(k))\dots(e_{t(k)}(k)), \end{cases} \tag{9}$$

where $a, k, e_i(k), t \in \mathbb{N}^+$, is called a *succession rule*; $(k) \rightsquigarrow (e_1(k))(e_2(k))\dots(e_{t(k)}(k))$ is a (possibly finite) set of *productions*, starting from (a) which is the *axiom*. Succession rules are familiar in literature [3, 4, 15, 17, 18, 19, 30, 31], and closely related to the ECO method.

Moreover the concept of generating tree can be naturally defined for any kind of succession rule: the root of the tree has label (a), and each node with label (k) has $t = t(k)$ sons with labels $(e_1(k))$, $(e_2(k))$, ..., $(e_t(k))$.

In particular, for any given succession rule Γ a non-decreasing sequence $\{u_n\}_{n\geq 0}$ of positive integers is then defined, u_n being the number of nodes at level n in the generating tree defined by Γ. By convention, the root is at level 0, so $u_0 = 1$. We also consider the generating function $u_\Gamma(x)$ of the sequence $\{u_n\}_{n\geq 0}$.

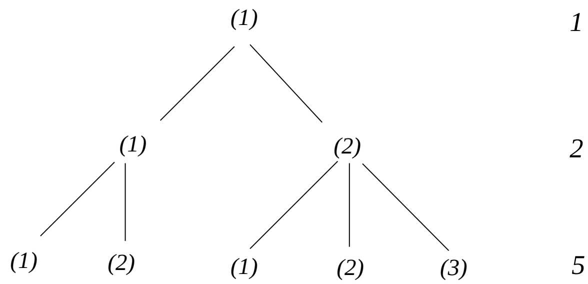

Figure 11: The first levels of the generating tree associated with the succession rule Γ.

For example, let Γ be the following succession rule (studied in various papers, but first presented in [4]):

$$\Gamma \left\{ \begin{array}{l} (1) \\ (k) \rightsquigarrow (1)(2)\dots(k+1), \end{array} \right. \tag{10}$$

defining a generating tree whose first levels are shown in Fig. 11. The reader can check that the rule defines the sequence of Catalan numbers.

Remark 2. Please note that for the remainder of the paper we will use the following notation, unless otherwise stated:

- $f(x)$ (resp. $g(x)$) is the generating function in (2) (resp. (4)) for the class of convex (resp. column-convex) polyominoes according to the semi-perimeter;
- $\{f_n\}_{n\geq 0}$ (resp. $\{g_n\}_{n\geq 0}$) is the sequence defined by $f(x)$ (resp. $g(x)$);
- Ω (resp. Δ) is the succession rule associated with the ECO operator for the class of convex (resp. column-convex) polyominoes.
- $f_\Omega(x)$ (resp. $g_\Delta(x)$) is the generating function of the succession rule Ω (resp. Δ); in practice, the functions $f(x)$ and $f_\Omega(x)$ are related by the simple relation:

$$f(x) = x^2 f_\Omega(x).$$

  Analogously, $g(x) = x^2 g_\Delta(x)$.

2.2 An ECO operator and a succession rule for column-convex polyominoes

In this section we present an ECO operator for the class $\mathcal{CC}$ of column-convex polyominoes. Being this construction quite similar to that proposed for convex polyominoes, we will here outline just the main features.

First we decompose $\mathcal{CC}$ into three mutually disjoint subclasses, and to do this we take into consideration the following conditions:

Q1 : the ordinate of the highest cell of the rightmost column is greatest than the ordinate of the highest cell of the column on its left;

Q2 : the ordinate of the highest cell of the rightmost column is equal to the ordinate of the highest cell of the column on its left;

Q3 : the ordinate of the lowest cell of the rightmost column is minor than or equal to the ordinate of the lowest cell of the column on its left.

Basing on these three conditions we define the following classes:

i. $\mathcal{CC}_b$ is the subclass of $\mathcal{CC}$ made of those polyominoes that satisfy both conditions **Q2** and **Q3** (see Fig.12, (b)).

ii. $\mathcal{CC}_g$ is the subclass of $\mathcal{CC}$ made of those polyominoes that do not satisfy any of the conditions **Q1**, **Q2** and **Q3** (see Fig.12, (g)).

iii. $\mathcal{CC}_a$ contains all polyominoes in $\mathcal{CC}$ that satisfy at least one of the conditions **Q1**, **Q2** and **Q3**, and do not lie in $\mathcal{CC}_b$ (Figure 12, (a), depicts three possible cases). In practice, a polyomino in $\mathcal{CC}_a$ can satisfy: only condition **Q1**, only condition **Q2**, only **Q3**, or both conditions **Q1** and **Q3**.

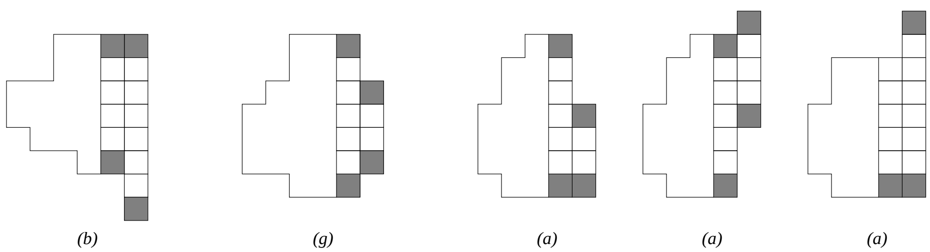

Figure 12: Column-convex polyominoes of the classes (b), (g), and (a).

Again, a polyomino of the class $\mathcal{CC}_i$, $i \in \{a, b, g\}$, with k cells in the rightmost column can be represented by the label $(k)_i$.

The operator on column-convex polyominoes, which we call ϑ_1, acts differently on polyominoes belonging to different classes. Its performance is similar to that of ϑ on convex polyominoes, therefore instead of giving a formal definition we prefer to give a graphical description, in Fig. 13.

The construction of the operator ϑ_1 can be easily be represented by means of the succession rule Δ:

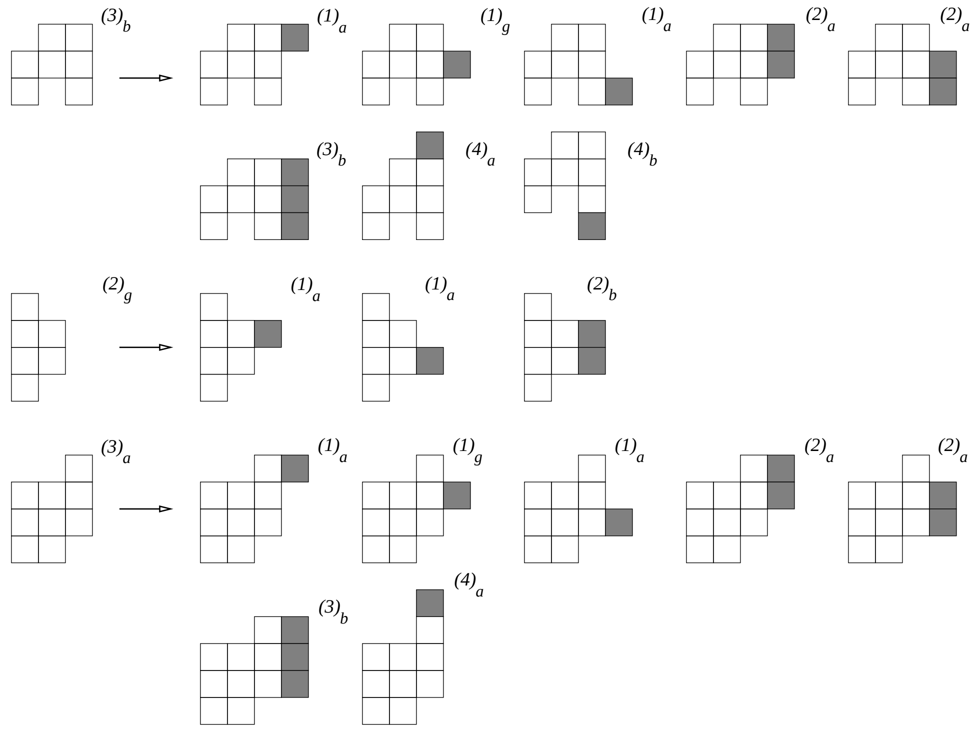

Figure 13: The performance of ϑ_1 on polyominoes of the classes (b), (g), and (a).

$$\left\{\begin{array}{l} (1)_a \\ \\ (k)_g \rightsquigarrow \prod_{j=1}^{k-2}(j)_g^{k-j-1}\ \prod_{j=1}^{k-1}(j)_a^2\ (k)_b \\ \\ (k)_a \rightsquigarrow \prod_{j=1}^{k-2}(j)_g^{k-j-1}\ \prod_{j=1}^{k-1}(j)_a^2\ (k)_b\ (k+1)_a \\ \\ (k)_b \rightsquigarrow \prod_{j=1}^{k-2}(j)_g^{k-j-1}\ \prod_{j=1}^{k-1}(j)_a^2\ (k)_b\ (k+1)_a\ (k+1)_b. \end{array}\right. \tag{11}$$

For instance, we have the following productions for $k = 3$:

$$(3)_g \rightsquigarrow (1)_g(1)_a(1)_a(2)_a(2)_a(3)_b,$$

$$(3)_a \rightsquigarrow (1)_g(1)_a(1)_a(2)_a(2)_a(3)_b(4)_a,$$

$$(3)_b \rightsquigarrow (1)_g(1)_a(1)_a(2)_a(2)_a(3)_b(4)_a(4)_b.$$

Remark 3. The reader can easily verify that *directed-convex polyominoes* are those in the class $\mathcal{CC}\backslash\mathcal{CC}_b$. Therefore, restricting the operator ϑ_1 to this class, we easily obtain an ECO construction (a pictorial description is given in Fig. 14; please notice that parallelogram polyominoes have labels $(k)_r$, while the remaining directed-convex polyominoes have label $(k)_g$, $k \geq 1$).

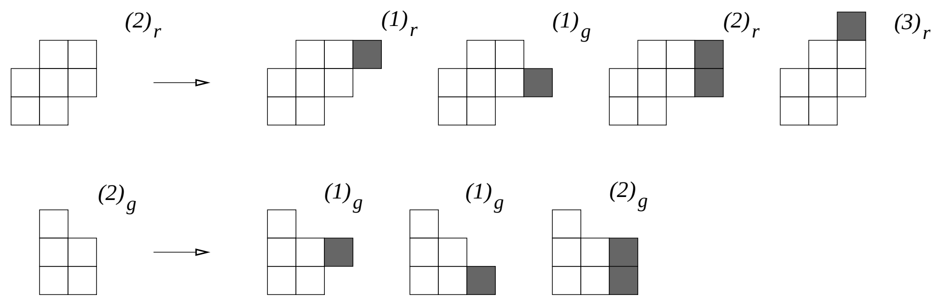

Figure 14: The ECO operator ϑ restricted to the class of directed-convex polyominoes.

The succession rule associated with this construction is then:

$$\begin{cases} (1)_a \\ \\ (k)_g \rightsquigarrow \prod_{j=1}^{k-2} (j)_g^{k-j-1} \ \prod_{j=1}^{k-1} (j)_a^2 \ (k)_a \\ \\ (k)_a \rightsquigarrow \prod_{j=1}^{k-2} (j)_g^{k-j-1} \ \prod_{j=1}^{k-1} (j)_a^2 \ (k)_a \ (k+1)_a . \end{cases} \qquad (12)$$

3 Determining the generating function of convex and column-convex polyominoes

Our next goal is to determine the generating function $f(x)$ of convex polyominoes according to the semi-perimeter, using the construction given by the ECO operator ϑ, and the associated succession rule Ω. We first achieve this purpose in Section 3.1 by applying methods provided in [3] to obtain the generating function of a generic succession rule. Then, in Section 3.2, we start afresh on a second, novel approach: we represent the nodes of the generating tree of the rule as terms of a formal power series, and then determine a recursive decomposition for this series. It is then easy to pass from such a decomposition to an algebraic system of equations satisfied by the generating function of the succession rule Ω.

The case of column-convex polyominoes is then treated with the same tools in Section 3.3.

3.1 The standard approach

We begin by translating the construction yielded by the operator ϑ into a functional equation which is satisfied by the generating function of convex polyominoes. Since this approach uses standard techniques, we will only sketch the main steps.

Let P be a convex polyomino. We denote the semi-perimeter and the number of cells in the rightmost column of P by $p(P)$ and $r(P)$, respectively. The bivariate generating function of the class $\mathcal{C}$ of convex polyominoes, according to these parameters is:

$$f(s,x)=\sum_{P\in\mathcal{C}} s^{r(P)}x^{p(P)}.$$

The generating functions of the classes $\mathcal{C}_a,\mathcal{C}_b,\mathcal{C}_g,\mathcal{C}_r$ are respectively:

$$A(s,x)=\sum_{P\in\mathcal{C}_a} s^{r(P)}x^{p(P)}, \qquad B(s,x)=\sum_{P\in\mathcal{C}_b} s^{r(P)}x^{p(P)},$$
$$G(s,x)=\sum_{P\in\mathcal{C}_g} s^{r(P)}x^{p(P)}, \qquad R(s,x)=\sum_{P\in\mathcal{C}_r} s^{r(P)}x^{p(P)}.$$

By determining $A(s,x), B(s,x), G(s,x)$ and $R(s,x)$, we get the desired generating function, $f(s,x)=A(s,x)+B(s,x)+G(s,x)+R(s,x)$. Most of the calculation that follows has been performed using MAPLE.

We remark that it is possible to refine the following calculation in order to consider also other parameters, such as the number of rows and columns, and the area.

Translating the construction defined by ϑ onto $A(s,x), B(s,x)$ yields:

$$A(s,x)=s\,x^2+s\,x\,A(s,x)+s\,x\,B(s,x), \quad B(s,x)=x\,A(s,x)+(1+s)\,x\,B(s,x).$$

So,

$$A(s,x)=\frac{x^2\,(1-x-s\,x)\,s}{1-x-2\,s\,x+s^2\,x^2}, \qquad B(s,x)=\frac{s\,x^3}{1-x-2\,s\,x+s^2\,x^2}. \tag{13}$$

Translating the construction onto $R(s,x)$ yields:

$$R(s,x)=\frac{2\,x}{1-s}\,\left(s\,A(1,x)-A(s,x))+s\,B(1,x)-B(s,x)\right)+\frac{s\,x}{1-s}\,\left(R(1,x)-sR(s,x)\right).$$

Using the generating functions (13), we have that:

$$R(s,x)=2\,\frac{(1-s\,x+s\,x^2)\,s\,x^4}{(1-3\,x+x^2)\,(1-x-2\,s\,x+s^2\,x^2)}+\frac{s\,x}{1-s}\,\left(R(1,x)-s\,R(s,x)\right),$$

and then

$$R(s,x)(1-s+s^2x)=2\,\frac{(1-s)(1-s\,x+s\,x^2)\,s\,x^4}{(1-3\,x+x^2)\,(1-x-2\,s\,x+s^2\,x^2)}+s\,x\,R(1,x). \tag{14}$$

A solution of this equation can be obtained by plainly applying the kernel method [3]; first we set the coefficient of $R(s,x)$ to be equal to 0,

$$(1-s+s^2x)=0,$$

obtaining the two solutions:

$$s_1 = \frac{1-\sqrt{1-4x}}{2x}, \qquad s_2 = \frac{1+\sqrt{1-4x}}{2x}.$$

Since only the first one is a well-defined power series, we perform the substitution $s = s_1$ in the equation (14), and then obtain the solution

$$R(s,x) = 2\,\frac{sx^4\,(3-2\,s\,x-\sqrt{1-4\,x})}{\sqrt{1-4\,x}\,(1-2\,s\,x+\sqrt{1-4\,x})\,(1-x-2\,s\,x+s^2\,x^2)}. \tag{15}$$

Translating the construction to $G(s,x)$ yields:

$$\begin{aligned}
&\frac{x}{(1-s)^2}\big(A(s,x)+B(s,x)+s\,R(s,x)+s^2\,G(s,x)\big)+\\
&\frac{s\,x}{(1-s)}\left(\frac{\partial}{\partial s}A(s,x)\Big|_{s=1}+\frac{\partial}{\partial s}B(s,x)\Big|_{s=1}+s\,\frac{\partial}{\partial s}R(s,x))\Big|_{s=1}+\frac{\partial}{\partial s}G(s,x)\Big|_{s=1}\right)+\\
&\frac{s\,x}{(1-s)^2}\left((s-2)\,A(s,x)+(s-2)\,B(s,x)-R(1,x)-s\,G(1,x)\right)=G(s,x).
\end{aligned}$$

Some parts of the previous summation have already been determined, in (13), (15), so in order to simplify the calculus we introduce the function $q(s,x)$, defined as:

$$\begin{aligned}
q(s,x) \quad = \quad &\frac{x}{(1-s)^2}\left(A(s,x)+B(s,x)+s\,R(s,x)\right)+\\
&\frac{s\,x}{(1-s)}\left(\frac{\partial}{\partial s}A(s,x)\Big|_{s=1}+\frac{\partial}{\partial s}B(s,x)\Big|_{s=1}+s\,\frac{\partial}{\partial s}R(s,x))\Big|_{s=1}\right)+\\
&\frac{s\,x}{(1-s)^2}\left((s-2)\,A(s,x)+(s-2)\,B(s,x)-R(1,x)\right).
\end{aligned}$$

For simplicity we omit the expression of $q(s,x)$. Performing some algebraic manipulations we obtain:

$$G(s,x) = q(s,x)+\frac{s^2\,x}{(1-s)^2}\,G(s,x)+\frac{s\,x}{(1-s)}\frac{\partial}{\partial s}\,G(s,x)\Big|_{s=1}-\frac{s^2\,x}{(1-s)^2}\,G(1,x). \tag{16}$$

We remark that in [6], Bousquet-Mélou encountered the same functional equation. We then write (16) as

$$G(s,x)\,\left((1-s)^2-s^2x\right) = q(s,x)(1-s)^2\,+\,sx(1-s)\frac{\partial}{\partial s}\,G(s,x)\Big|_{s=1}-\,s^2x\,G(1,x), \tag{17}$$

and solve the *kernel*

$$(1-s)^2-s^2x=0,$$

obtaining the two solutions:

$$s_1 = \frac{1-\sqrt{x}}{1-x}, \qquad s_2 = \frac{1+\sqrt{x}}{1-x}.$$

We remark that both s_1 and s_2 are well-defined formal power series. Substituting first s_1 and then s_2 in (17), we obtain two equations, where the left side equals zero, and the unknowns are $G(1,x)$ and $\frac{\partial}{\partial s} G(s,x)\Big|_{s=1}$. The solutions are:

$$\begin{aligned}\frac{\partial}{\partial s} G(s,x)\Big|_{s=1} \;=\;& \frac{(-2+26\,x-132\,x^2+330\,x^3-421\,x^4+253\,x^5-61\,x^6)\,x}{1-14\,x+75\,x^2-190\,x^3+225\,x^4-104\,x^5+16\,x^6} \\ & -\frac{(10\,x^6-182\,x^3+92\,x^2-22\,x+2+172\,x^4-70\,x^5)\,x(1-\sqrt{4x})}{1-14\,x+75\,x^2-190\,x^3+225\,x^4-104\,x^5+16\,x^6},\end{aligned}$$

$$\begin{aligned}G(1,x) \;=\;& -\frac{(4\,x^4-20\,x^3+30\,x^2-14\,x+2)\,x^2\sqrt{1-4x}}{1-11\,x+41\,x^2-56\,x^3+16\,x^4} \\ & -\frac{(4\,x^5-23\,x^4+59\,x^3-54\,x^2+18\,x-2)\,x^2}{1-11\,x+41\,x^2-56\,x^3+16\,x^4}.\end{aligned}$$

At this stage we have determined all the terms needed to compute $f(1,x)$. Finally,

$$f(1,x) \;=\; A(1,x) \;+\; B(1,x) \;+\; R(1,x) \;+\; G(1,x),$$

where $A(1,x)$, $B(1,x)$, and $R(1,x)$ are easily obtained from (13) and (15):

$$f(x) = f(1,x) = x^2\,\frac{\left(1-6\,x+11\,x^2-4\,x^3-4\,x^2\sqrt{1-4\,x}\right)}{1-8\,x+16\,x^2},$$

which is the desired result.

3.2 A new approach

The approach in Section 3.1, while establishing that the generating function for convex polyominoes indexed by semi-perimeter is indeed *algebraic*, still leaves the fact that this is so something of a mystery. In the present section, our aim is to find the generating function $f_\Omega(x)$ of the rule Ω (and therefore $f(x)$) by a different approach. To this end, we rely on the idea, introduced by Fédou and Garcia, in [17], of working on succession rules by means of non-commutative formal power series.

Each convex polyomino is uniquely identified by a node N of the generating tree of the rule Ω, and this node can be encoded by a word in the infinite alphabet $\Sigma = \{(i)_a, (j)_b, (h)_g, (l)_r \,:\, i,j,h,l \in \mathbb{N}^+\}$. Such a word can be understood in an obvious sense as the sequence of labels of the nodes in the path starting from the root and ending at N. As an example, the polyomino depicted in Fig. 15 is encoded by the word $(1)_a(1)_b(2)_b(3)_a(3)_b(4)_a(5)_a(3)_r(2)_g(2)_g$.

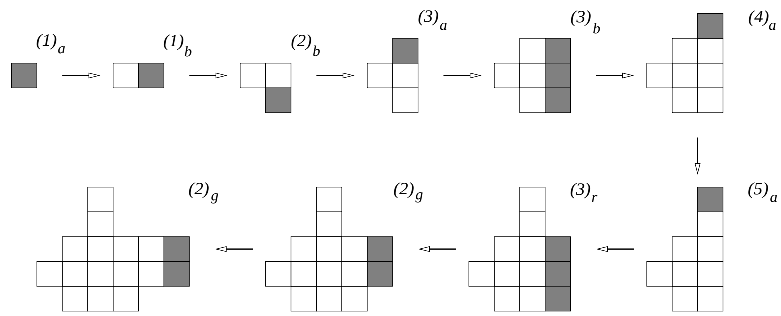

Figure 15: The ECO construction of a convex polyomino and the corresponding word.

Because of the form of the productions of the rule Ω, some convex polyominoes have necessarily the same word representation. For example the word $(1)_a(2)_a(1)_r$ represents two polyominoes of size 4, as the reader can easily check looking at Fig. 10.

The considerations made in the previous lines can be suitably stated in a more formal way. Let L_Ω be the set of words, over Σ, beginning with $(1)_a$ and satisfying the productions of Ω. Each word w of L_Ω corresponds to at least one path in the generating tree of Ω. We denote by S_Ω the noncommutative formal power series:

$$S_\Omega = \sum_{w \in L_\Omega} m(w)w,$$

where $m(w)$ is the number of paths corresponding to w in the generating tree of Ω. By construction, the generating function $S_\Omega(x)$ of the series S_Ω, and $f(x)$ are related by,

$$f(x) = xS_\Omega(x).$$

For example, we have

$$S_\Omega = (1)_a + (1)_a(1)_b + (1)_a(2)_a + (1)_a(1)_b(1)_b + (1)_a(1)_b(2)_a + (1)_a(1)_b(2)_b +$$

$$2 \cdot (1)_a(2)_a(1)_r + (1)_a(2)_a(2)_b + (1)_a(2)_a(3)_a + \dots$$

$$S_\Omega(x) = x + 2x^2 + 7x^3 + 28x^4 + 122x^5 + 558x^6 + \dots$$

We work on the series S_Ω using the standard operations on noncommutative formal power series; in particular, for any positive integer n, and $(i)_j \in \Sigma$:

$$nS_\Omega = \sum_{w \in L_\Omega} (nm(w))\, w, \qquad (i)_j S_\Omega = \sum_{w \in L_\Omega} m(w)(i)_j w.$$

Using the same notation of [17], we introduce the operation $\oplus$: for any word of L_Ω,

$$u = (i_1)_{j_1}(i_2)_{j_2} \dots (i_k)_{j_k}, \text{ we set}$$

$$u^{\oplus} = (i_1+1)_{j_1}(i_2+1)_{j_2}\dots(i_k+1)_{j_k}.$$

For example $((1)_a(2)_a(1)_r)^{\oplus} = (2)_a(3)_a(2)_r$. Moreover:

$$L_{\Omega}^{\oplus} = \left\{ w^{\oplus} \,:\, w \in L_{\Omega} \right\}, \qquad \text{and} \qquad S_{\Omega}^{\oplus} = \sum_{w \in L_{\Omega}} (m(w)) w^{\oplus}.$$

It is a neat consequence that $S_{\Omega}^{\oplus}$ and S_{Ω} have the same generating function.

Generally speaking, a noncommutative formal power series S_{Γ}, and its generating function $S_{\Gamma}(x)$ can be associated with any succession rule Γ in a completely analogous way.

Catalan succession rule. To fully understand the heart of the matter, we start presenting an example already given in [17]. Let us consider succession rule defining Catalan numbers, already presented in Section 2:

$$\Gamma \left\{ \begin{array}{l} (1) \\ (k) \rightsquigarrow (1)(2)\dots(k+1), \end{array} \right. \tag{18}$$

Let $C = S_{\Gamma}$ be the noncommutative formal power series associated with the words of Γ. In practice:

$$C = (1) + (1)(1) + (1)(2) + (1)(1)(1) + (1)(1)(2) + (1)(2)(1) + (1)(2)(2) + (1)(2)(3) + \dots$$

Easily, we prove that:

$$C = (1) + (1)\ C + (1)\ C^{\oplus} + (1)\ C^{\oplus}\ C. \tag{19}$$

In fact, let w be a term of C. If $|w| \neq 1$, then $w = (1)v$, with $|v| \geq 1$. So we have these possibilities:

1) v begins with (1). Then w is a term of the series $(1)C$.

2) $v = (2)z$, with $z = (u_1)\dots(u_k)$, and $u_i > 1$, for $i \in \{1,\dots,k\}$. In this case $(2)z$ is a term of $C^{\oplus}$, and then w is a term of $(1)C^{\oplus}$.

3) $v = (2)(u_1)\dots(u_k)w_2$, where $u_i > 1$, for $i \in \{1,\dots,k\}$, and w_2 begins with (1). Then, w is a term of $(1)C^{\oplus}C$.

The equation (19) provides a recursive decomposition of the series C, from which we immediately derive a functional equation satisfied by the generating function $C(x)$:

$$C(x) = x + x\ C(x) + x\ C(x) + x\ C^2(x). \tag{20}$$

The basic idea to deal with the succession rule Ω for convex polyominoes is substantially the same, but it requires some more precautions.

A succession rule defining central binomial coefficients. In order to ensure that all the steps in our approach are more readily comprehensible, we present in the following a detailed description of the calculus of the generating function for the succession rule previously determined in (12), which is indeed more complex than (10).

Let us recall that the rule (12), that for brevity we will call Ω', has the same productions as Ω, but the axiom is $(1)_r$ instead of $(1)_a$. In practice:

$$\Omega' \begin{cases} (1)_r \\ \\ (k)_g \rightsquigarrow \prod_{j=1}^{k} (j)_g^{k-j+1} \\ \\ (k)_r \rightsquigarrow \prod_{j=1}^{k-1} (j)_g^{k-j} \ \prod_{j=1}^{k+1} (j)_r. \end{cases} \tag{21}$$

In this paragraph our aim is to give a proof that the succession rule Ω' defines the sequence of central binomial coefficients, $\binom{2n}{n}$. We also advise the reader that to determine the generating function of Ω', rather than being a mere exercise, will remarkably simplify the computation of $f_\Omega(x)$.

As usual, let us denote by $L_{\Omega'}$ the set of the words produced by Ω', and let $R = S_{\Omega'} = \sum_{w \in L_{\Omega'}} m(w)w$. The main theorem is preceded by two technical lemmas, easily provable by induction.

Lemma 3.1 In the succession rule Ω', the label $(k_2)_{j_2}$ is produced by $(k_1)_{j_1}$ if and only if $(k_2-1)_{j_2}$ is produced by $(k_1-1)_{j_1}$, with $k_1, k_2 > 1$, and $j_1, j_2 \in \{g, r\}$.

Using Lemma 3.1 we are able to prove the following.

Lemma 3.2 Let $L_P = \{u : u = (2)_r (u_2)_{j_2} \dots (u_k)_{j_k},\ u_i > 1\,,\ \text{for } i \in \{2, \dots k\},\ \text{and } (1)_r u \in L_{\Omega'}\}$. Then $L_P = L^{\oplus}_{\Omega'}$.

Proof.

($\Rightarrow$) Let $u = (2)_r (u_2)_{j_2} \dots (u_k)_{j_k} \in L_P$. By definition of $L^{\oplus}_{\Omega'}$, $u \in L^{\oplus}_{\Omega'}$ if $u^{\ominus} = (1)_r (u_2 - 1)_{j_2} \dots (u_k - 1)_{j_k} \in L_{\Omega'}$. We proceed by induction on the length of $u^{\ominus}$.

Base: if $|u^{\ominus}| = 1$ the result immediately follows;

Step $n \to n+1$: let $u = (2)_r (u_2)_{j_2} \dots (u_n)_{j_n} (u_{n+1})_{j_{n+1}} \in L_P$. By inductive hypothesis, the word $(1)_r (u_2 - 1)_{j_2} \dots (u_n - 1)_{j_n}$ belongs to $L_{\Omega'}$. By Lemma 3.1, the label $(u_{n+1} - 1)_{j_{n+1}}$ is produced by the label $(u_n - 1)_{j_n}$ according to the productions of the rule Ω'. Consequently $u^{\ominus} \in L_{\Omega'}$.

($\Leftarrow$) The result can be achieved again by induction.

□

Theorem 3.1 The noncommutative formal power series R can be decomposed into the following sum:

$$R = (1)_r + (1)_r R + (1)_r R^{\oplus} + (1)_r C^{\oplus} R + (1)_r P^{\oplus} G + (1)_r Q^{\oplus} G, \tag{22}$$

where

$$\begin{aligned} C = \quad & (1)_r + (1)_r C + (1)_r C^{\oplus} + (1)_r C^{\oplus}\ C \\ G = \quad & (1)_g + (1)_g G \\ P = \quad & (1)_r + (1)_r P + (1)_r C^{\oplus} P + (1)_r P^{\oplus} + (1)_r C^{\oplus} \\ Q = \quad & (1)_r Q + (1)_r C^{\oplus} Q + 2(1)_r P^{\oplus} G + 2(1)_r Q^{\oplus} G + \\ & (1)_r Q^{\oplus}\ +\ (1)_r (R - C)^{\oplus}. \end{aligned} \tag{23}$$

Proof. In order to let the reader have a better comprehension of the role of each term of the sum in (22), we give in Fig. 16 a rough representation of the generating tree of Ω'.

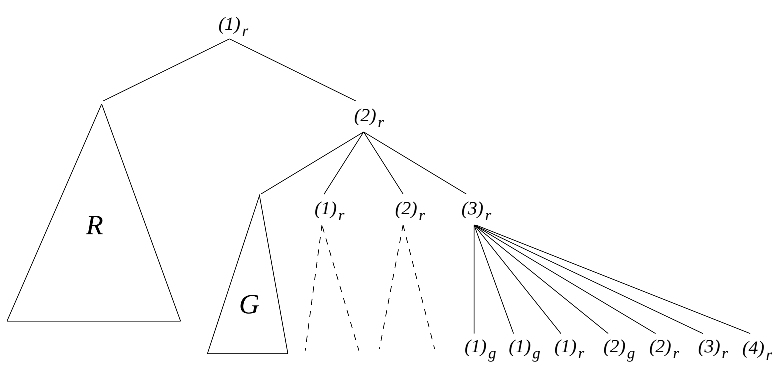

Figure 16: The first levels of the generating tree of Ω'.

Let w be a term of the series $L_{\Omega'}$. The following cases may occur:

-$|w| = 1$, then $w = (1)_r$.

- $|w| > 1$ then $w = (1)_r v$, and we distinguish the following cases:

 1) v begins with $(1)_r$. The set of words in $L_{\Omega'}$ having the form $w = (1)_r v$ is then equal to $(1)_r L_{\Omega'}$. Consequently

$$\sum_{w=(1)_r v} m(w) w = (1)_r R.$$

 For example, the word $(1)_r(1)_r(1)_r(2)_r(1)_r(1)_r(2)_r(3)_r(1)_r$ is a term of $(1)_r R$.

2) v begins with $(2)_r$. As sketched in Fig. 16, four cases are possible:

a) $v \in L_P = \{u : u = (2)_r (u_2)_{j_2} \dots (u_k)_{j_k},\ u_i > 1\,,$ for $i \in \{2, \dots k\}$ and $(1)_r u \in L_{\Omega'}\}$.
Since, from Lemma 3.2, $L_P = L^{\oplus}_{\Omega'}$, we have

$$\sum_{w=(1)_r v,\, v \in L_P} m(w)w = (1)_r \sum_{v \in L_P} m(v)v = (1)_r R^{\oplus},$$

For example, the word $(1)_r(2)_r(3)_r(4)_r(5)_r(3)_g(3)_g(2)_g$ is a term of $(1)_r R^{\oplus}$.

b) $v = (2)_r (u_2)_r \dots (u_k)_r (1)_r w_2$, with $u_i > 1$ for $i \in \{1, \dots k\}$, and $(1)_r w_2 \in L_{\Omega'}$.
The reader can easily check that the language of such words having the form $(2)_r (u_2)_r \dots (u_k)_r$, $u_i > 1$, coincides with $L^{\oplus}_{\Gamma}$, i.e. the series that we have previously named $C^{\oplus}$, associated with Catalan numbers:

$$\sum_w m(w)w = (1)_r \sum_{v \in L^{\oplus}_{\Gamma} L_{\Omega'}} m(v)v = (1)_r C^{\oplus} R.$$

The word $(1)_r(2)_r(3)_r(2)_r(1)_r(2)_r(3)_r(2)_g(2)_g(1)_g$ is a term of $(1)_r C^{\oplus} R$.

c) $v = (2)_r (u_2)_r \dots (u_k)_r g_1$, where $u_i > 1$, for $i \in \{2, \dots, k\}$, and g_1 is a sequence of labels $(1)_g$, i.e. an element of

$$L_G = \{(1)_g, (1)_g(1)_g, (1)_g(1)_g(1)_g, (1)_g(1)_g(1)_g(1)_g, \dots\}.$$

Using considerations similar to those in step *b*), the sum over all words w of this type leads to:

$$\sum_w m(w)w = (1)_r \sum_{v \in L^{\oplus}_{\Gamma} L_G} m(v)v. \tag{24}$$

In this case, for any word v, the value $m(v)$ depends on u_k, the term preceding g_1; more precisely, according to the productions of Ω', and letting $u_k = j$,

$$m(v) = (j-1) \cdot m\left((2)_r (u_2)_r \dots (j)_r\right) \cdot m(g_1). \tag{25}$$

For $i \geq 1$ let us denote by $L_{\Gamma(i)}$ the set of the words of L_{Γ} ending with $(i)_t \in \Sigma$, $t \in \{a, b, r, g\}$. Using equations (24), and (25), the sum over the words w of such form leads to:

$$\sum_w m(w)w = (1)_r \sum_{j \geq 2} \left((j-1) \cdot \sum_{v \in L^{\oplus}_{\Gamma(j-1)}} m(v)v \right) \sum_{v \in L_G} m(v)v.$$

Letting

$$C_{(j-1)} = \sum_{v \in L_{\Gamma}(j-1)} m(v)v \quad \text{and} \quad G = \sum_{v \in L_G} m(v)v,$$

equation (24) becomes

$$\sum_w m(w)w = (1)_r \sum_{j\geq 2}(j-1)C^{\oplus}_{(j-1)}G = (1)_r P^{\oplus}G,$$

where we have set $P = \sum_{j\geq 2}(j-1)C_{(j-1)}$.
The word $(1)_r(2)_r(3)_r(4)_r(2)_r(3)_r(1)_g(1)_g(1)_g$ is an example of a term of $(1)_r P^{\oplus}G$.

d) $v = (2)_r(u_2)_{j_2}\dots(u_k)_{j_k}g_1$, with $u_i > 1$, for $i \in \{2,\dots,k\}$, $j_k = g$, and $g_1 \in L_G$. Considerations analogous to those in step *c*) suggest that the sum over all words w of this type leads to $(1)_r Q^{\oplus}G$,
where

$$Q = \sum_{j\geq 2} j\ R_{(j-1)_g} \text{ and } R_{(j-1)_g} = \sum_{v\in L_{\Omega'(j-1)_g}} m(v)v,$$

$L_{\Omega'(j-1)_g}$ being the set of words belonging to $L_{\Omega'}$ and ending with $(j-1)_g$.
The word $(1)_r(2)_r(3)_r(4)_r(2)_r(3)_r(2)_g(2)_g(1)_g(1)_g$ is an example of a term of $(1)_r Q^{\oplus}G$.

It is easy to verify that the decomposition (23) takes into account all the words that satisfy the succession rule Ω', and each one is computed with the exact multiplicity.

To conclude the proof we must verify that the noncommutative formal power series C, G, P, and Q satisfy the system of equations (23). The statement is obvious for C and G. Below, we will prove that P satisfies:

$$P = (1)_r + (1)_r P + (1)_r C^{\oplus}P + (1)_r P^{\oplus} + (1)_r C^{\oplus}, \tag{26}$$

recalling that $P = \sum_{j\geq 2}(j-1)C_{(j-1)}$. From the first equation in (23) we deduce that:

$$\begin{aligned} C_{(i)} =& \quad (1)_r\ C_{(i)} + (1)_r\ C^{\oplus}_{(i-1)} + (1)_r\ C^{\oplus}\ C_{(i)} \quad \text{for } i > 1, \\ C_{(1)} =& \quad (1)_r + (1)_r\ C_{(1)} + (1)_r\ C^{\oplus}\ C_{(1)}. \end{aligned} \tag{27}$$

Consequently

$$\begin{aligned} P &= C_{(1)} + \textstyle\sum_{j\geq 3}(j-1)C_{(j-1)} \\ &= (1)_r + (1)_r\ C_{(1)} + (1)_r\ C^{\oplus}\ C_{(1)} + (1)_r \textstyle\sum_{j\geq 3}(j-1)\ C_{(j-1)} \\ &+ (1)_r \textstyle\sum_{j\geq 3}(j-1)\ C^{\oplus}_{(j-2)} + (1)_r \textstyle\sum_{j\geq 3}(j-1)\ C^{\oplus}\ C_{(j-1)}. \end{aligned}$$

By performing simple algebraic manipulations we obtain (26).

A similar argument leads to the decomposition of Q. Let us recall that $Q = \sum_{j\geq 2} j\ R_{(j-1)_g}$.
From the equation for R we deduce that:

$$R_g = (1)_r R_g + (1)_r R^{\oplus}_g + (1)_r C^{\oplus} R_g + (1)_r P^{\oplus} G + (1)_r Q^{\oplus} G, \tag{28}$$

consequently

$$\begin{aligned} R_{(i)_g} = &\quad (1)_r\ R_{(i)_g} + (1)_r\ R^{\oplus}_{(i-1)_g} + (1)_r C^{\oplus} R_{(i)_g} &\qquad \text{for } i > 1, \\ R_{(1)_g} = &\quad (1)_r R_{(1)_g} + (1)_r\ C^{\oplus}\ R_{(1)_g} + (1)_r P^{\oplus} G + (1)_r Q^{\oplus} G. \end{aligned} \tag{29}$$

Then

$$\begin{aligned} Q &= 2\ R_{(1)_g} + \textstyle\sum_{j\geq 3} j\ R_{(j-1)_g} \\ &= 2(1)_r R_{(1)_g} + 2(1)_r\ C^{\oplus}\ R_{(1)_g} + 2(1)_r P^{\oplus} G + 2(1)_r Q^{\oplus} G + \\ &\quad + (1)_r \textstyle\sum_{j\geq 3} j\ R_{(j-1)_g} + (1)_r \sum_{j\geq 3} j R^{\oplus}_{(j-2)_g} + (1)_r C^{\oplus} \sum_{j\geq 3} j R_{(j-1)_g}. \end{aligned}$$

By performing some algebraic manipulations we obtain that

$$\begin{aligned} Q &= (1)_r Q + (1)_r C^{\oplus} Q + 2(1)_r P^{\oplus} G + 2(1)_r Q^{\oplus} G + (1)_r \textstyle\sum_{j\geq 3} j R^{\oplus}_{(j-2)_g} \\ &= (1)_r Q + (1)_r C^{\oplus} Q + 2(1)_r P^{\oplus} G + 2(1)_r Q^{\oplus} G + (1)_r \textstyle\sum_{j\geq 2} j R^{\oplus}_{(j-1)_g} + (1)_r \sum_{j\geq 2} R^{\oplus}_{(j-1)_g} \\ &= (1)_r Q + (1)_r C^{\oplus} Q + 2(1)_r P^{\oplus} G + 2(1)_r Q^{\oplus} G + (1)_r Q^{\oplus} + (1)_r\ (R - C)^{\oplus}. \end{aligned}$$

□

From (23) we obtain a system of functional equations,

$$\begin{aligned} R(x) = &\quad x + x\ R(x) + x\ R(x) + x\ C(x)\ R(x) + x\ P(x)\ G(x) + \\ &\quad x\ Q(x)\ G(x) \\ C(x) = &\quad x + x\ C(x) + x\ C(x) + x\ C^2(x) \\ G(x) = &\quad x + x\ G(x) \\ P(x) = &\quad x + x\ P(x) + x\ C(x)\ P(x) + x\ P(x) + x\ C(x) \\ Q(x) = &\quad x\ Q(x) + x\ C(x)\ Q(x) + 2x\ P(x)\ G(x) + 2x\ Q(x)\ G(x)\ + \\ &\quad xQ(x)\ +\ x(R(x) - C(x)). \end{aligned} \tag{30}$$

Finally, from (30) we derive the generating function of Ω',

$$R(x) = \frac{x}{\sqrt{1-4x}}.$$

We remark that the decomposition given in (22) is effectively a context-free unambiguous grammar generating the class of directed-convex polyominoes.

3.2.1 The decomposition for the rule Ω

Let A be the noncommutative formal power series associated with the succession rule Ω for convex polyominoes, i.e.

$$A = S_\Omega = \sum_{w \in L_\Omega} m(w)w.$$

Using the same method applied in the previous case we manage to determine a decomposition for the series A, and then translate it into a system of equations. Before let us define some other formal power series:

- Ω'' is the succession rule having the same productions as Ω and starting with the axiom $(1)_b$, and
$$B = \sum_{w \in L_{\Omega''}} m(w)w;$$
- for any non-empty subset χ of $\{a, b, r, g\}$, and $i \geq 1$, let
$$L_{\Omega(i)_\chi} = \cup_{q\in\chi} L_{\Omega(i)_q} \qquad L_{\Omega''(i)_\chi} = \cup_{q\in\chi} L_{\Omega''(i)_q},$$
where $L_{\Omega(i)_q}$ (resp. $L_{\Omega''(i)_q}$) denotes the set of words of L_Ω (resp. $L_{\Omega''}$) ending with $(i)_q$, $i \geq 0$. Moreover, let $A_{(i)_\chi}$ (resp. $B_{(i)_\chi}$) be the noncommutative formal power series associated with $L_{\Omega(i)_\chi}$ (resp. $L_{\Omega''(i)_\chi}$);
- for any non-empty subset χ of $\{a, b, r, g\}$, let $L_{\Omega\chi} = \cup_{q\in\chi} L_{\Omega q}$ (resp. $L_{\Omega''\chi} = \cup_{q\in\chi} L_{\Omega'' q}$), where $L_{\Omega q}$ (resp. $L_{\Omega'' q}$) denotes the sets of words of L_Ω (resp. $L_{\Omega''}$) ending with any label of type q; finally A_χ (resp. B_χ) denotes the corresponding formal power series.

The following technical lemmas are easily provable by observing the productions of Ω and Ω''.

Lemma 3.3 Let us consider the succession rules Ω and Ω''; the label $(k_2)_{j_2}$ is produced by $(k_1)_{j_1}$ if and only if $(k_2 - 1)_{j_2}$ is produced by $(k_1 - 1)_{j_1}$, with $k_1, k_2 > 1$, $j_1, j_2 \in \{a, b, g, r\}$.

Lemma 3.4 Let $L_Q = \{u = (2)_a (u_2)_{j_2} \dots (u_k)_{j_k} \,|\, u_i > 1\,,\text{ for } i \in \{2, \dots k\}, \text{ and } (1)_a u \in L_\Omega\}$, and $L_R = \{u = (2)_b (u_2)_{j_2} \dots (u_k)_{j_k} \,|\, u_i > 1\,,\text{ for } i \in \{2, \dots k\}, \text{ and } (1)_b u \in L_{\Omega''}\}$. We have:

i) $L_Q = L_\Omega^\oplus$;

ii) $L_R = L_{\Omega''}^\oplus$.

Theorem 3.2 The noncommutative formal power series A can be decomposed into the following sum:

$$\begin{aligned} A \;=\;& (1)_a \;+\; (1)_a\, B \;+\; (1)_a\, A^{\oplus} \;+\; 2(1)_a\, A^{\oplus}_{a,b}\, R\,+ \\ & (1)_a\, A^{\oplus}_{r}\, R \;+\; (1)_a\, P^{\oplus}_{A}\, G \;+\; (1)_a\, Q^{\oplus}_{A}\, G \;+\; (1)_a\, S^{\oplus}_{A}\, G, \end{aligned} \tag{31}$$

where

$$\begin{aligned} B \;=\;& \xi(A) \;+\; (1)_b\, B^{\oplus} \;+2(1)_b\;\, B^{\oplus}_{a,b}\, R +\; (1)_b\, B^{\oplus}_{r}\, R\,+ \\ & (1)_b\, P^{\oplus}_{B} G + (1)_b\, Q^{\oplus}_{B} G \;+\; (1)_b\, S^{\oplus}_{B} G, \end{aligned} \tag{32}$$

$$\begin{aligned} A_{a,b} =\;& (1)_a + (1)_a\, B_{a,b} + (1)_a\, A^{\oplus}_{a,b} \\ B_{a,b} =\;& \xi(A_{a,b}) + (1)_b\, B^{\oplus}_{a,b} \\ A_r =\;& (1)_a\, B_r + (1)_a\, A^{\oplus}_{r} + 2(1)_a\, A^{\oplus}_{a,b}\, C + (1)_a\, A^{\oplus}_{r}\, C \\ B_r =\;& \xi(A_r) + (1)_b\, B^{\oplus}_{r} + 2(1)_b\, B^{\oplus}_{a,b}\, C + (1)_b\, B^{\oplus}_{r}\, C, \\ A_g =\;& (1)_a\, B_g + (1)_a\, A^{\oplus}_{g} + 2(1)_a\, A^{\oplus}_{a,b}\, R_g + (1)_a\, A^{\oplus}_{r}\, R_g\, + \\ & (1)_a\, P^{\oplus}_{A}\, G + (1)_a\, Q^{\oplus}_{A}\, G + (1)_a\, S^{\oplus}_{A}\, G \\ B_g =\;& \xi(A_g) + (1)_b\, B^{\oplus}_{g} + 2(1)_b\, B^{\oplus}_{a,b}\, R_g + (1)_b\, B^{\oplus}_{r}\, R_g\, + \\ & (1)_b\, P^{\oplus}_{B}\, G + (1)_b\, Q^{\oplus}_{B}\, G + (1)_b\, S^{\oplus}_{B}\, G, \end{aligned} \tag{33}$$

$$\begin{aligned} P_A =\;& (1)_a\, P_B + (1)_a\, P^{\oplus}_{A} + (1)_a\, A^{\oplus}_{a,b} \\ P_B =\;& \xi(P_A) + (1)_b\, P^{\oplus}_{B} + (1)_b\, B^{\oplus}_{a,b} \\ Q_A =\;& (1)_a\, Q_B + 2(1)_a\, A^{\oplus}_{a,b}\, P + (1)_a\, A^{\oplus}_{r}\, P + (1)_a\, Q^{\oplus}_{A} + (1)_a\, A^{\oplus}_{r} \\ Q_B =\;& \xi(Q_A) + 2(1)_b\, B^{\oplus}_{a,b}\, P + (1)_b\, B^{\oplus}_{r}\, P + (1)_b\, Q^{\oplus}_{B} + (1)_b\, B^{\oplus}_{r} \\ S_A =\;& (1)_a\, S_B + 2(1)_a\, A^{\oplus}_{a,b}\, Q + (1)_a\, A^{\oplus}_{r}\, Q + 2(1)_a\, P^{\oplus}_{A}\, G\; + 2(1)_a\, Q^{\oplus}_{A}\, G\, + \\ & 2(1)_a\, S^{\oplus}_{A}\, G\; + (1)_a\, S^{\oplus}_{A} + (1)_a\, A^{\oplus}_{g} \\ S_B =\;& \xi(S_A) + 2(1)_b\, B^{\oplus}_{a,b}\, Q + (1)_b\, B^{\oplus}_{r}\, Q + 2(1)_b\, P^{\oplus}_{B}\, G + 2(1)_b\, Q^{\oplus}_{B}\, G\, + \\ & 2(1)_b\, S^{\oplus}_{B}\, G + (1)_b\, S^{\oplus}_{B} + (1)_b\, B^{\oplus}_{g}, \end{aligned} \tag{34}$$

where R, G, P, Q, and R_g are the formal power series defined and calculated for the rule Ω', and, for any formal power series Z on the alphabet Σ, $\xi(Z)$ is defined as the formal power series obtained from Z by replacing, in each term, the first occurrence of $(1)_a$ with $(1)_b$.

Proof. Our first step is to compute the equations for A and B. Figure 17 shows a rough decomposition of the generating trees of Ω and Ω''.

The equation for A**.** Let w be a word of L_Ω. Then we have the following cases:

-$|w| = 1$, then $w = (1)_a$.

- $|w| > 1$ then $w = (1)_a v$, and we distinguish the following cases:

1) v begins with $(1)_b$. The set of words in L_Ω having the form $w = (1)_a v$ is then equal to $(1)_a L_{\Omega''}$. Consequently

$$\sum_{w \in (1)_a L_{\Omega''}} m(w)w = (1)_a B.$$

For example, the word $(1)_a(1)_b(2)_a(1)_r(2)_r(3)_r(2)_g(2)_g(1)_g$ is a term of $(1)_a B$.

2) v begins with $(2)_a$. Six cases are possible:

a) $v \in L_Q$, where L_Q is defined in Lemma 3.4, and there it is shown that $L_Q = L_\Omega^\oplus$. Consequently

$$\sum_{w \in (1)_a L_\Omega{}^\oplus} m(w)w = (1)_a \sum_{v \in L_Q} m(v)v = (1)_a\, A^\oplus.$$

The word $(1)_a(2)_a(3)_a(3)_b(2)_r(3)_r(2)_g$ is a term of $(1)_a\, A^\oplus$.

b) $v = (2)_a(u_2)_{j_2} \dots (u_{k-1})_{j_{k-1}}(u_k)_{j_k}(1)_r w_2$, with $u_i > 1$ for $i \in \{1, \dots, k\}$ and $j_k \in \{a, b\}$. The set of words in L_Ω having the form $(1)_a\, v$, is then equal to $(1)_a\, L_{\Omega}{}^\oplus_{a,b} L_{\Omega'}$. Consequently, summing over all the words of such a type, we have:

$$\sum_w m(w)w = (1)_a \sum_{v \in L_{\Omega}{}^\oplus_{a,b} L_{\Omega'}} m(v)v = 2(1)_a\, A^\oplus_{a,b} R.$$

For example, the word $(1)_a(2)_a(3)_a(3)_b(3)_b(4)_a(1)_r(2)_r(1)_g(1)_g$ is a term of $(1)_a\, A^\oplus_{a,b} R$.

c) $v = (2)_a(u_2)_{j_2} \dots (u_k)_r(1)_r w_2$, with $u_i > 1$ for $i \in \{1, \dots k\}$. The set of words in L_Ω having the form $(1)_a\ v$, is then equal to $(1)_a\ L_{\Omega}{}^\oplus_r L_{\Omega'}$. Consequently, summing over all the words of such a type, we have:

$$\sum_w m(w)w = (1)_a \sum_{v \in L_{\Omega}{}^\oplus_r L_{\Omega'}} m(v)v = (1)_a\, A^\oplus_r R.$$

The word $(1)_a(2)_a(3)_a(3)_b(3)_b(2)_r(1)_r(2)_r(1)_g(1)_g$ is a term of $(1)_a\, A^\oplus_r R$.

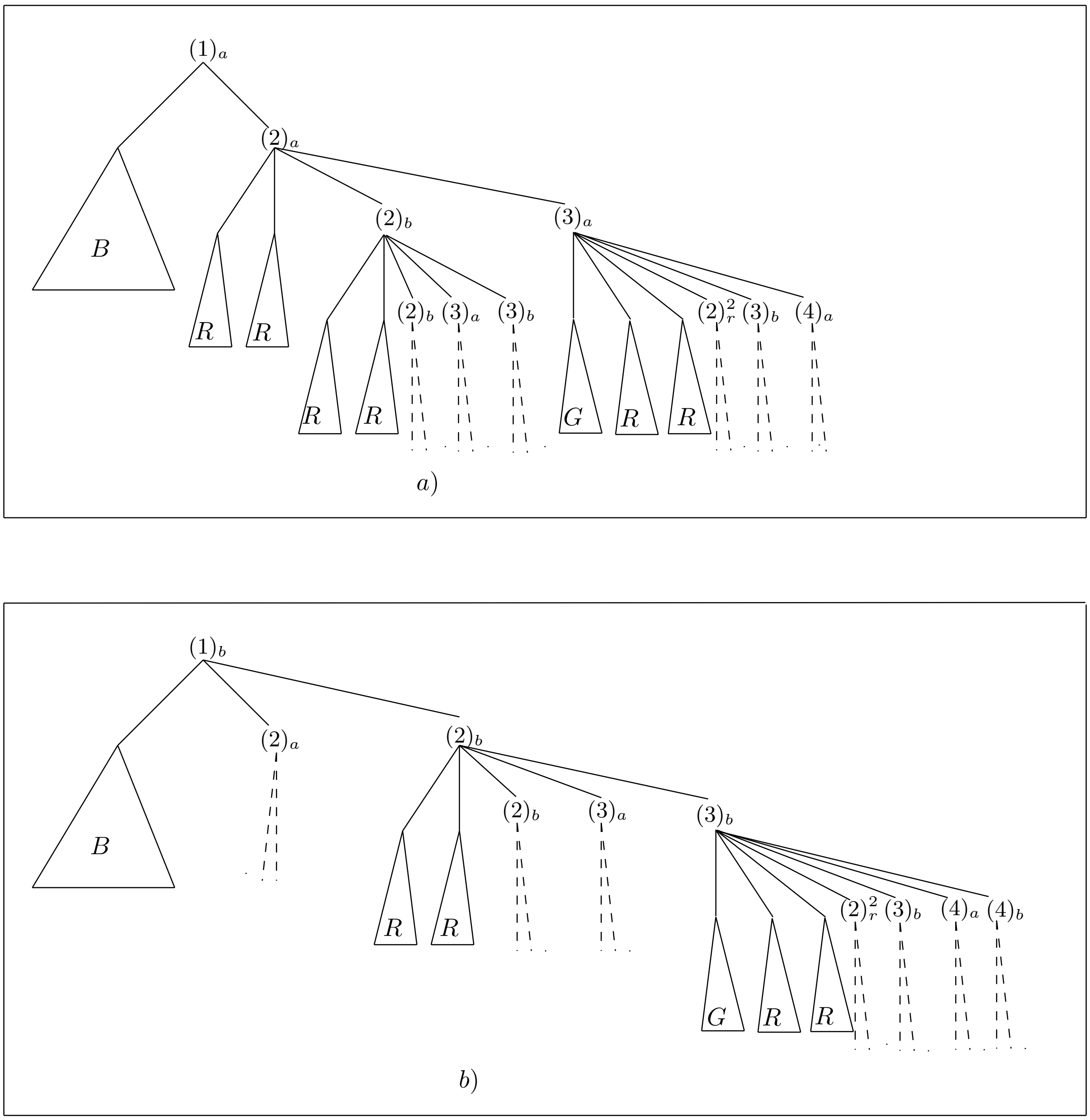

Figure 17: The first levels of the generating trees associated with the rules Ω (a), and Ω'' (b).

d) $v=(2)_a(u_2)_{j_2}\dots(u_k)_{j_k}g_1$, where $u_i>1$, for $i\in\{2,\dots,k\}$, $j_k\in\{a,b\}$ and $g_1\in L_G=\{(1)_g,(1)_g(1)_g,(1)_g(1)_g(1)_g,\dots\}$. Then, summing over all the words of such a type, we have:

$$\sum_w m(w)w=(1)_a\sum_{v\in L_{\Omega^\oplus_{a,b}}L_G} m(v)v.$$

In this case the value $m(v)$ also depends on the value u_k: if $u_k=j+1$ then, according to the rule Ω,

$$m(v)=(j-1)\,m\left((2)_a(u_2)_{j_2}\dots(j+1)_{j_k}\right)\,m(g_1).$$

Then we have:

$$\sum_w m(w)w=(1)_a\sum_{j\geq 2}(j-1)\sum_{v\in L_{\Omega^\oplus_{(j)_{a,b}}}} m(v)v\sum_{v\in L_G} m(v)v,$$

and consequently

$$\sum_w m(w)w=(1)_a\sum_{j\geq 2}(j-1)\;A^\oplus_{(j)_{a,b}}G.$$

By denoting $P_A=\sum_{j\geq 2}(j-1)\;A_{(j)_{a,b}}$ we have

$$\sum_w m(w)w=(1)_a\;P_A^\oplus G.$$

For example, the word $(1)_a(2)_a(3)_a(3)_b(4)_b(1)_g(1)_g$ is a term of $(1)_a\;P_A^\oplus G$.

e) $v=(2)_a(u_2)_{j_2}\dots(u_k)_r g_1$, where $u_i>1$, for $i\in\{2,\dots,k\}$, and $g_1\in L_G$. Then we have

$$\sum_w m(w)w=(1)_a\sum_{v\in L_{\Omega^\oplus_r}L_G} m(v)v.$$

Also in this case the value $m(v)$ depends on the value u_k: if $u_k=j$ then, according to the rule Ω,

$$m(v)=(j-1)\,m\left((2)_a(u_2)_{j_2}\dots(j)_r\right)\,m(g_1).$$

Then we have:

$$\sum_w m(w)w=(1)_a\sum_{j\geq 2}(j-1)\sum_{v\in L_{\Omega^\oplus_{(j-1)_r}}} m(v)v\sum_{v\in L_G} m(v)v,$$

and consequently

$$\sum_w m(w)w=(1)_a\sum_{j\geq 2}(j-1)\;A^\oplus_{(j-1)_r}G.$$

By denoting $Q_A = \sum_{j\geq 2}(j-1)\ A_{(j-1)_r}$ we have

$$\sum_w m(w)w = (1)_a\ Q_A^{\oplus} G.$$

For instance, the word $(1)_a(2)_a(3)_a(3)_b(4)_b(2)_r(1)_g$ is a term of $(1)_a\ Q_A^{\oplus} G$.

f) $v = (2)_a(u_2)_{j_2}\ldots(u_k)_g g_1$, where $u_i > 1$, for $i \in \{2,\ldots,k\}$, and $g_1 \in L_G$. The set of words in L_Ω having the form $(1)_a\ v$, is then equal to $(1)_a\ L_{\Omega}{}_g^{\oplus}\ L_G$. Consequently

$$\sum_w m(w)w = (1)_a \sum_{v\in L_{\Omega}{}_g^{\oplus}\ L_G} m(v)v.$$

The value $m(v)$ also depends on the value u_k: if $u_k = j$ then, according to the rule Ω,

$$m(v) = (j)\ m\left((2)_a(u_2)_{j_2}\ldots(j)_g\right)\ m(g_1). \tag{35}$$

Then we have:

$$\sum_w m(w)w = (1)_a \sum_{j\geq 2}(j) \sum_{v\in L_{\Omega}{}_{(j-1)_g}^{\oplus}} m(v)v \sum_{v\in L_G} m(v)v,$$

and consequently

$$\sum_w m(w)w = (1)_a \sum_{j\geq 2} j\ A_{(j-1)_g}^{\oplus} G.$$

By denoting $S_A = \sum_{j\geq 2} j\ A_{(j-1)_g}$ we have

$$\sum_w m(w)w = (1)_a\ S_A^{\oplus} G.$$

The word $(1)_a(2)_a(3)_a(3)_b(4)_b(2)_g(1)_g(1)_g$ is a term of $(1)_a\ S_A^{\oplus} G$.

The equation for B. In order to determine a decomposition for the series B, we first need to introduce the auxiliary language $L_{\Omega'''}$, constituted by those words in L_Ω where the first occurrence, i.e. the letter $(1)_a$, is replaced by $(1)_b$. In practice,

$$L_{\Omega'''} = \left\{(1)_b u\ :\ (1)_a u \in L_\Omega\right\}.$$

Analogously, let $\xi(A)$ be the formal power series obtained from A by replacing, in each term, the first occurrence of $(1)_a$ with $(1)_b$. Then $\xi(A)$ is the formal power series of the words of $L_{\Omega'''}$.

Let $w \in L_{\Omega''}$. Then $w \in L_{\Omega'''}$ or $w = (1)_b v$, where v begins with $(2)_b$ (see Figure 17). This last one is therefore the only case left to examine. Then six cases are possible:

a) $v \in L_R$, L_R being defined in Lemma 3.4. From Lemma 3.4 it holds that $L_R = L^{\oplus}_{\Omega''}$. Consequently

$$\sum_w m(w)w = (1)_b \sum_{v \in L_R} m(v)v = (1)_b\ B^{\oplus}.$$

For example, the word $(1)_b(2)_b(2)_b(3)_b(4)_a(3)_r(2)_g$ is a term of $(1)_b\ B^{\oplus}$.

b) $v = (2)_b(u_2)_{j_2}\dots(u_k)_{j_k}(1)_r w_2$, with $u_i > 1$ for $i \in \{1,\dots k\}$ and $j_k \in \{a,b\}$. The set of words in L''_Ω having the form $(1)_b\ v$, is then equal to $(1)_b\ L^{\oplus}_{\Omega''_{a,b}}\ L_{\Omega'}$. Consequently

$$\sum_w m(w)w = (1)_b \sum_{v \in L_{\Omega''}{}^{\oplus}_{a,b}\ L_{\Omega'}} m(v)v = 2(1)_b\ B^{\oplus}_{a,b}R.$$

The word $(1)_b(2)_b(3)_b(4)_a(3)_b(1)_r(2)_r(1)_g$ is a term of $(1)_b\ B^{\oplus}_{a,b}R$.

c) $v = (2)_b(u_2)_{j_2}\dots(u_k)_r(1)_r w_2$, with $u_i > 1$ for $i \in \{1,\dots k\}$. The set of words in $L_{\Omega''}$ having the form $(1)_b\ v$, is then equal to $(1)_b\ L_{\Omega''}{}^{\oplus}_{r}\ L_{\Omega'}$. Consequently

$$\sum_w m(w)w = (1)_b \sum_{v \in L_{\Omega''}{}^{\oplus}_{r}\ L_{\Omega'}} m(v)v = (1)_b\ B^{\oplus}_r R.$$

The word $(1)_b(2)_b(3)_b(4)_a(3)_b(4)_a(3)_r(1)_r(2)_r(1)_g$ is a term of $(1)_b\ B^{\oplus}_r R$.

d) $v = (2)_b(u_2)_{j_2}\dots(u_k)_{j_k} g_1$, where $u_i > 1$, for $i \in \{2,\dots,k\}$, $j_k \in \{a,b\}$ and $g_1 \in L_G$. By using the same arguments as in the calculus for A we obtain

$$\sum_w m(w)w = (1)_b \sum_{j\geq 2} (j-1) \sum_{v \in L_{\Omega''}{}^{\oplus}_{(j)_{a,b}}} m(v)v \sum_{v \in L_G} m(v)v =$$

$$(1)_b \sum_{j\geq 2}(j-1)\ B^{\oplus}_{(j)_{a,b}} G = (1)_b\ P^{\oplus}_B G,$$

where $P_B = \sum_{j\geq 2} (j-1)\ B_{(j)_{a,b}}$. The word $(1)_b(2)_b(3)_b(4)_a(3)_b(1)_g(1)_g$ is a term of $(1)_b\ \ P^{\oplus}_B G$.

e) $v = (2)_b(u_2)_{j_2}\dots(u_k)_r g_1$, where $u_i > 1$, for $i \in \{2,\dots,k\}$, and $g_1 \in L_G$. In practice,

$$\sum_w m(w)w = (1)_b \sum_{j\geq 2} (j-1) \sum_{v \in L_{\Omega''}{}^{\oplus}_{(j-1)_r}} m(v)v \sum_{v \in L_G} m(v)v =$$

$$(1)_b \sum_{j\geq 2}(j-1)\ B^{\oplus}_{(j-1)_r} G = (1)_b\ Q^{\oplus}_B G,$$

where Q_B denotes the series $\sum_{j\geq 2}(j-1)\ B_{(j-1)_r}$.

For example, the word $(1)_b(2)_b(3)_b(4)_a(3)_b(4)_a(3)_r(1)_g$ is a term of $(1)_b\ \ Q^{\oplus}_B G$.

$f)$ $v=(2)_b(u_2)_{j_2}\dots(u_k)_g g_1$, where $u_i>1$, for $i\in\{2,\dots,k\}$, and $g_1\in L_G$.

$$\sum_w m(w)w=(1)_b\sum_{j\ge 2}(j)\sum_{v\in L_{\Omega''}{}^{\oplus}_{(j-1)_g}} m(v)v\sum_{v\in L_G} m(v)v,$$

$$(1)_b\sum_{j\ge 2} j\ B^{\oplus}_{(j-1)_g}G=(1)_b\ S^{\oplus}_B G,$$

where $S_B=\sum_{j\ge 2} j\ B_{(j-1)_g}$.

For example, the word $(1)_b(2)_b(3)_b(4)_a(3)_b(4)_a(3)_r(2)_g(1)_g(1)_g$ is a term of $(1)_b\ \ S^{\oplus}_B G$.

To conclude the proof we must verify that the equation in (33) are satisfied by $A_{a,b}$, $B_{a,b}$, A_r, B_r, A_g, and B_g. In practice, all the equations in (33) can be easily deduced from the equations (31) and (32) for the series A and B. For instance the equation for $A_{a,b}$ is obtained from the equation for A, with the precaution that the terms ending with R and G do not contribute to $A_{a,b}$.

Concerning (34), here we compute the equations for P_A, Q_A, and S_A, omitting for brevity's sake the calculus of P_B, Q_B, and S_B, which is analogous. Let us recall that $P_A=\sum_{j\ge 2}(j-1)\ A_{(j)_{a,b}}$. From the equation for $A_{a,b}$, in (33), we have

$$A_{(i)_{a,b}}=(1)_a\ B_{(i)_{a,b}}+(1)_a\ A^{\oplus}_{(i-1)_{a,b}}\qquad \text{for } i>1.$$

Then by substituting the expression for $A_{(j)_{a,b}}$ we obtain

$$\begin{aligned}P_A&=(1)_a\sum_{j\ge 2}(j-1)\ B_{(j)_{a,b}}+(1)_a\sum_{j\ge 2}(j-1)A^{\oplus}_{(j-1)_{a,b}}\\&=(1)_a\ P_B+(1)_a\ A^{\oplus}_{(1)_{a,b}}+(1)_a\sum_{j\ge 3}(j-1)A^{\oplus}_{(j-1)_{a,b}}\\&=(1)_a\ P_B+(1)_a\ A^{\oplus}_{a,b}+(1)_a\ P^{\oplus}_A.\end{aligned}$$

Now, we compute $Q_A=\sum_{j\ge 2}(j-1)\ A_{(j-1)_r}$. From the equation for A_r, in (33), we have

$$\begin{aligned}A_{(i)_r}&=(1)_a\ B_{(i)_r}+(1)_a\ A^{\oplus}_{(i-1)_r}+2(1)_a\ A^{\oplus}_{a,b}\ C_{(i)}+(1)_a\ A^{\oplus}_r\ C_{(i)}\qquad \text{if } i>1,\\A_{(1)_r}&=(1)_a\ B_{(1)_r}++2(1)_a\ A^{\oplus}_{a,b}\ C_{(1)}+(1)_a\ A^{\oplus}_r\ C_{(1)}.\end{aligned}\tag{36}$$

By substituting $A_{(j-1)_r}$ in Q_A we have

$$\begin{aligned}Q_A=\ &A_{(1)_r}+(1)_a\ \sum_{j\ge 3}(j-1)\ \ B_{(j-1)_r}+(1)_a\ \sum_{j\ge 3}(j-1)A^{\oplus}_{(j-2)_r}+\\&+2(1)_a\ A^{\oplus}_{a,b}\sum_{j\ge 3}(j-1)\ C_{(j-1)}+(1)_a\ A^{\oplus}_r\sum_{j\ge 3}(j-1)\ C_{(j-1)}.\end{aligned}$$

Then, by substituting $A_{(1)_r}$ we have

$$\begin{aligned} Q_A \; &= (1)_a\, Q_B + 2(1)_a\, A^{\oplus}_{a,b}\, P + (1)_a\, A^{\oplus}_r\, P + (1)_a\, \sum_{j\geq 3}(j-1)A^{\oplus}_{(j-2)_r} \\ &= (1)_a\, Q_B + +2(1)_a\, A^{\oplus}_{a,b}\, P + (1)_a\, A^{\oplus}_r\, P + (1)_a Q^{\oplus}_A + (1)_a\, A^{\oplus}_r \end{aligned}$$

The previous considerations lead us to finally determine S_A. Indeed, from the equation for A_g in (33) we have

$$\begin{aligned} A_{(i)_g} = &\quad (1)_a\, B_{(i)_g} + (1)_a\, A^{\oplus}_{(i-1)_g} + 2(1)_a\, A^{\oplus}_{a,b}\, R_{(i)_g} + (1)_a\, A^{\oplus}_r\, R_{(i)_g} \quad \text{if } i > 1, \\ A_{(1)_g} = &\quad (1)_a\, B_{(1)_g} + 2(1)_a\, A^{\oplus}_{a,b}\, R_{(1)_g} + (1)_a\, A^{\oplus}_r\, R_{(1)_g} + (1)_a\, P^{\oplus}_A\, G + \\ &\quad (1)_a\, Q^{\oplus}_A\, G + (1)_a\, S^{\oplus}_A\, G. \end{aligned} \tag{37}$$

We recall that $S_A = \sum_{j\geq 2} j\, A_{(j-1)_g}$. Then, by substituting $A_{(j-1)_g}$ we obtain

$$\begin{aligned} S_A = &\;\; 2A_{(1)_g} + (1)_a \sum_{j\geq 3} j\, B_{(j-1)_g} + (1)_a\, \sum_{j\geq 3} j A^{\oplus}_{(j-2)_g} + \\ &\;\; 2(1)_a\, A^{\oplus}_{a,b}\, \sum_{j\geq 3} j\, R_{(j-1)_g} + (1)_a\, A^{\oplus}_r\, \sum_{j\geq 3} j\, R_{(j-1)_g}. \end{aligned}$$

Then, by substituting $A_{(1)_g}$ we have

$$\begin{aligned} S_A \;\; = &\;\; (1)_a\, S_B + (1)_a\, \sum_{j\geq 3} j A^{\oplus}_{(j-2)_g} + 2(1)_a\, A^{\oplus}_{a,b}\, Q + (1)_a\, A^{\oplus}_r\, Q + \\ &\;\; +2(1)_a\, P^{\oplus}_A\, G\, + 2(1)_a\, Q^{\oplus}_A\, G + 2(1)_a\, S^{\oplus}_A\, G. \\ = &\;\; (1)_a\, S_B + 2(1)_a\, A^{\oplus}_{a,b}\, Q + (1)_a\, A^{\oplus}_r\, Q + 2(1)_a\, P^{\oplus}_A\, G + \\ &\;\; +2(1)_a\, Q^{\oplus}_A\, G + 2(1)_a\, S^{\oplus}_A\, G + (1)_a\, S^{\oplus}_A + (1)_a\, A^{\oplus}_g. \end{aligned}$$

□

Finally, by taking the commutative image of the equations of the system obtained in (31), (32), (33), and (34), we have a system of functional equations and we can compute the generating function $A(x)$ of the series A. The system that we obtain is the following, where the expressions for $R(x)$, $C(x)$, $P(x)$, $Q(x)$, and $G(x)$ have already been determined.

$$\begin{aligned} A(x) \;=\; &\;\; x \;+\; x\, B(x) \;+\; x\, A(x) \;+\; 2x\, A_{a,b}(x)\, R(x) \;+\; x\, A_r(x)\, R(x) + \\ &\;\; x\, P_A(x)\, G(x) \;+\; x\, Q_A(x)\, G(x) \;+\; x\, S_A(x)\, G(x), \end{aligned} \tag{38}$$

where

$$\begin{aligned} B(x) \;=\; &\;\; A(x) \;+\; x\, B(x) \;+ 2x\;\; B_{a,b}(x)\, R(x) +\; x\, B_r(x)\, R(x) + \\ &\;\; x\, P_B(x) G(x) + x\, Q_B(x) G(x) \;+\; x\, S_B(x) G(x), \end{aligned} \tag{39}$$

$$
\begin{aligned}
A_{a,b}(x) = &\quad x + x\ B_{a,b}(x) + x\ A_{a,b}(x)\\
B_{a,b}(x) = &\quad A_{a,b}(x) + x\ B_{a,b}(x)\\
A_r(x) = &\quad x\ B_r(x) + x\ A_r(x) + 2x\ A_{a,b}(x)\ C(x) + x\ A_r(x)\ C(x)\\
B_r(x) = &\quad A_r(x) + x\ B_r(x) + 2x\ B_{a,b}(x)\ C(x) + x\ B_r(x)\ C(x),\\
A_g(x) = &\quad x\ B_g(x) + x\ A_g(x) + 2x\ A_{a,b}(x)\ R_g(x) + x\ A_r(x)\ R_g(x) +\\
&\quad x\ P_A(x)\ G(x) + x\ Q_A(x)\ G(x) + x\ S_A(x)\ G(x)\\
B_g(x) = &\quad A_g(x) + x\ B_g(x) + 2x\ B_{a,b}(x)\ R_g(x) + x\ B_r(x)\ R_g(x) +\\
&\quad x\ P_B(x)\ G(x) + x\ Q_B(x)\ G(x) + x\ S_B(x)\ G(x)\\
R_g(x) = &\quad xR_g(x) + xR_g(x) + xC(x)R_g(x) + xP(x)G(x) + xQ(x)G(x),
\end{aligned}
\tag{40}
$$

$$
\begin{aligned}
P_A(x) = &\quad x\ P_B(x) + x\ P_A(x) + x\ A_{a,b}(x)\\
P_B(x) = &\quad P_A(x) + x\ P_B(x) + x\ B_{a,b}(x)\\
Q_A(x) = &\quad x\ Q_B(x) + 2x\ A_{a,b}(x)\ P(x) + x\ A_r(x)\ P(x) + x\ Q_A(x) + x\ A_r(x)\\
Q_B(x) = &\quad Q_A(x) + 2x\ B_{a,b}(x)\ P(x) + x\ B_r(x)\ P(x) + x\ Q_B(x) + x\ B_r(x)\\
S_A(x) = &\quad x\ S_B(x) + 2x\ A_{a,b}(x)\ Q(x) + x\ A_r(x)\ Q(x) + 2x\ P_A(x)\ G(x) +\\
&\quad 2x\ Q_A(x)\ G(x)\ + 2x\ S_A(x)\ G(x)\ + x\ S_A(x) + x\ A_g(x)\\
S_B(x) = &\quad S_A(x) + 2x\ B_{a,b}(x)\ Q(x) + x\ B_r(x)\ Q(x) + 2x\ P_B(x)\ G(x) +\\
&\quad 2x\ Q_B(x)\ G(x)\ + 2x\ S_B(x)\ G(x) + x\ S_B(x) + x\ B_g(x).
\end{aligned}
\tag{41}
$$

Many of the terms of the system are already known, and as a consequence the equations for $A_{a,b}(x)$, $B_{a,b}(x)$, $A_r(x)$, and $B_r(x)$ are linear. Once these series have been obtained, all the remaining equations are linear. Solving the system we get:

$$
f_\Omega(x) = \frac{A(x)}{x} = \sum_{n\geq 0} f_n x^n = \frac{1-6x+11x^2-4x^3-4x^2\sqrt{1-4x}}{(1-4x)^2}, \tag{42}
$$

then we determine the desired generating function for convex polyominoes,

$$
f(x) = x^2 f_\Omega(x).
$$

3.3 The enumeration of column-convex polyominoes

In the present section, we seek the generating function for the class of column-convex polyominoes, by appealing to the ECO construction for such a class defined in Section 2.2, and its associated succession rule Δ. In order to realize this aim, we use the same approach as for convex polyominoes. First, let us introduce the succession rules Δ' and Δ'', having the same productions as Δ, and starting with the axiom $(1)_b$ and $(1)_g$, respectively. By using the same arguments as for Ω (see Lemmas 3.3, 3.4), we are able to prove the following:

Lemma 3.5 In the succession rules Δ and Δ', the label $(k_2)_{j_2}$ is produced by $(k_1)_{j_1}$ if and only if $(k_2-1)_{j_2}$ is produced by $(k_1-1)_{j_1}$, with $k_1, k_2 > 1$.

Lemma 3.6 Let $L_Q = \{u = (2)_a(u_2)_{j_2}\dots(u_k)_{j_k} \mid u_i > 1\,,\text{ for } i \in \{2,\dots k\}\,,\text{ and } (1)_a u \in L_\Delta\}$, and $L_R = \{u = (2)_b(u_2)_{j_2}\dots(u_k)_{j_k} \mid u_i > 1\,,\text{ for } i \in \{2,\dots k\}, \text{ and } (1)_b u \in L_{\Delta'}\}$. Then we have:

i) $L_Q = L_\Delta^{\oplus}$;

ii) $L_R = L_{\Delta'}^{\oplus}$.

Let us denote by A, B, and G the non commutative formal power series associated respectively with the succession rules Δ, Δ', and Δ''. Then we have the following theorem:

Theorem 3.3 The noncommutative formal power series A can be decomposed into the following sum:

$$(1)_a + (1)_a B + (1)_a A^{\oplus} + 2(1)_a A^{\oplus} A + (1)_a P^{\oplus} G \tag{43}$$

where

$$\begin{aligned}
B = \quad & (1)_b + (1)_b B + (1)_b A^{\oplus} + 2(1)_b A^{\oplus} A + (1)_b P^{\oplus} G + \\
& (1)_b B^{\oplus} + 2(1)_b B^{\oplus} A + (1)_b Q^{\oplus} G \\
G = \quad & (1)_g + (1)_g B \\
P = \quad & (1)_a Q + (1)_a A^{\oplus} + (1)_a P^{\oplus} + 2(1)_a A^{\oplus} P + (1)_a P^{\oplus}(1)_g Q \\
Q = \quad & (1)_b Q + (1)_b A^{\oplus} + (1)_b P^{\oplus} + 2(1)_b A^{\oplus} P + (1)_b P^{\oplus}(1)_g Q + \\
& 2(1)_b B^{\oplus} P + (1)_b Q^{\oplus}(1)_g Q + (1)_b B^{\oplus} \ + \ (1)_b Q^{\oplus}.
\end{aligned} \tag{44}$$

Proof. The reader will immediately see that the equation for G is trivially verified. In order to make a simplification, we just verify the equations for A and P. The others are obtained by using the same technique.

The equation for A. Let w be a word of L_Δ, we have the following cases:

-$|w|=1$, then $w=(1)_a$.

- $|w|>1$ then $w=(1)_a v$, and we distinguish the following cases:

1) v begins with $(1)_b$. The set of words in L_Δ having the form $w=(1)_a v$ is then equal to $(1)_a L_{\Delta'}$. Consequently w is a term of the formal power series $(1)_a B$. For example, the word $(1)_a(1)_b(2)_a(2)_b(3)_b(1)_g$ is a term of $(1)_a B$.

2) v begins with $(2)_a$. Three cases are possible:

   a) $v \in L_Q$, where L_Q is defined in Lemma 3.6). Since we have proved that $L_Q = L_\Delta^\oplus$, then w is a term of $(1)_a \sum_{v \in L_Q} m(v)v = (1)_a\, A^\oplus$.
   For example, the word $(1)_a(2)_a(3)_a(3)_b(4)_a(2)_g$ is a term of $(1)_a\, A^\oplus$.

   b) $v=(2)_a(u_2)_{j_2}\ldots(u_k)_{j_k}(1)_a w_2$, with $u_i>1$ for $i\in\{1,\ldots k\}$. The set of words in L_Δ having the form $(1)_a\, v$, is then equal to $(1)_a\, L_\Delta^\oplus L_\Delta$.
   Consequently w is a term of the series $(1)_a \sum_{v\in L_\Delta^\oplus L_\Delta} m(v)v = 2\;(1)_a\, A^\oplus A$, where the coefficient 2 is explained by the multiplicity of the label $(1)_a$ in the rule Δ. For example, the word $(1)_a(2)_a(3)_a(3)_b(3)_b(1)_a(1)_b(2)_a(2)_b$ is a term of $(1)_a\, A^\oplus A$.

   c) $v=(2)_a(u_2)_{j_2}\ldots(u_k)_{j_k}(1)_g w_2$, where $u_i>1$, for $i\in\{2,\ldots,k\}$. Then w is a term of the series $(1)_a \sum_{v\in L_\Delta^\oplus L_{\Delta''}} m(v)v$. In this expression, $m(v)$ also depends on the value u_k: if $u_k=j+1$ then, according to the rule Δ,

   $$m(v) = (j-1)\; m\left((2)_a(u_2)_{j_2}\ldots(j+1)_{j_k}\right)\; m((1)_g w_2).$$

   Then w is a term of:

   $$(1)_a \textstyle\sum_{j\ge 2}(j-1)\sum_{v\in L_{\Delta_j}^\oplus} m(v)v\; \sum_{v\in L_{\Delta''}} m(v)v =$$

   $$(1)_a \textstyle\sum_{j\ge 2}(j-1)\; A_j^\oplus G = (1)_a\; P^\oplus G,$$

   where
   $$P = \sum_{j\ge 2}\,(j-1)\;A_j \tag{45}$$

   The word $(1)_a(2)_a(3)_a(3)_a(4)_a(1)_g(1)_b$ is a term of $(1)_a\; P^\oplus G$.

The equation for P. Let $Q=\sum_{j\ge 2}\,(j-1)\;A_j$. By replacing the equation for G in the equation (43) we deduce the following:

$$A_i = (1)_a\; B_i + (1)_a\; A_{i-1}^\oplus + 2(1)_a A^\oplus A_i + (1)_a P^\oplus (1)_g B_i \quad \text{for } i>1,$$

Then, by substituting A_j in the equation for P, we obtain

$$\begin{aligned} P &= (1)_a \sum_{j\geq 2} (j-1)\, B_j + (1)_a \sum_{j\geq 2} (j-1) A^{\oplus}_{(j-1)} + 2(1)_a A^{\oplus} \sum_{j\geq 2} (j-1) A_j + \\ &\quad + (1)_a P^{\oplus} (1)_g \sum_{j\geq 2} (j-1) B_j \\ &= (1)_a Q + (1)_a A^{\oplus}_1 + (1)_a \sum_{j\geq 3} (j-1) A^{\oplus}_{(j-1)} + 2(1)_a A^{\oplus} P + (1)_a P^{\oplus} (1)_g Q. \end{aligned}$$

By making some manipulations we obtain that

$$P = (1)_a Q + (1)_a A^{\oplus}_1 + (1)_a \sum_{j\geq 2} (j-1) A^{\oplus}_j + (1)_a \sum_{j\geq 2} A^{\oplus}_j + 2(1)_a A^{\oplus} P + (1)_a P^{\oplus} (1)_g Q$$

$$= (1)_a Q + (1)_a A^{\oplus} + (1)_a P^{\oplus} + 2(1)_a A^{\oplus} P + (1)_a P^{\oplus} (1)_g Q. \qquad \square$$

From Theorem 3.3 we immediately derive a system of functional equations,

$$\begin{aligned} A(x) =&\ \ x + xB(x) + xA(x) + 2xA^2(x) + xP(x)G(x) \\ B(x) =&\ \ x + xB(x) + xA(x) + 2xA^2(x) + xP(x)G(x) + \\ &\ \ xB(x) + 2xB(x)A(x) + xQ(x)G(x) \\ G(x) =&\ \ x + xB(x) \\ P(x) =&\ \ xQ(x) + xA(x) + xP(x) + 2xA(x)P(x) + x^2P(x)Q(x) \\ Q(x) =&\ \ xQ(x) + xA(x) + xP(x) + 2xA(x)P(x) + x^2P(x)Q(x) + \\ &\ \ 2xB(x)P(x) + x^2Q^2(x) + xB(x) \ + \ xQ(x). \end{aligned} \tag{46}$$

This system leads to the solution $A(x) = xg_{\Delta}(x) = g(x)/x$, and then finally we obtain the expression (4) for $g(x)$. We remark that (46) is the first example of a system of algebraic equations satisfied by the generating function of column-convex polyominoes.

4 Exhaustive generation using the ECO method

The recursive construction defined by the ECO operator ϑ, for the class of convex polyominoes, suggests a simple generation algorithm for this class. Indeed, in the generating tree associated with ϑ, each convex polyomino with semi-perimeter $n+2$ is uniquely identified by a path from the root to a node at level n (being the root is at level 0; see also Fig. 10).

Therefore, the generation of a convex polyomino of size $n+2$ consists in generating a path from the root to a node at level n of the generating tree.

4.1 A CAT algorithm for the generation of convex polyominoes

In this section we develop a CAT algorithm for generating all the convex polyominoes with given semi-perimeter $n+2$. The algorithm performs a prefix traversal in the first $n+1$ levels of the generating tree of the succession rule Ω.

We choose to represent each node $(k)_x$ as a pair (k, x) where $k \in \mathbb{N}^+$ is the label and $x \in \{a, b, g, r\}$ is the color. Consequently each sequence of labels having length $n+1$ can be represented by means of a matrix V of dimension $2\times(n+1)$ such that, for $1 \leq j \leq n+1$, the entries $V[j,1]$ and $V[j,2]$ contain the label and the color of the j-th term of the path, respectively.

We also remind the reader that, by convention, the sons of a node are ordered according to the productions of such a node defined by the succession rule (8).

As an example we list the seven matrices of size 2×3 associated with the set of convex polyominoes with semi-perimeter 4, and obtained by performing the traversal in the first 3 levels (see Fig. 10).

$$V_1 = \begin{pmatrix} 1 & 2 & 1 \\ a & a & r \end{pmatrix} \quad V_2 = \begin{pmatrix} 1 & 2 & 1 \\ a & a & r \end{pmatrix} \quad V_3 = \begin{pmatrix} 1 & 2 & 2 \\ a & a & a \end{pmatrix}$$

$$V_4 = \begin{pmatrix} 1 & 2 & 3 \\ a & a & a \end{pmatrix} \quad V_5 = \begin{pmatrix} 1 & 1 & 1 \\ a & b & b \end{pmatrix} \quad V_6 = \begin{pmatrix} 1 & 1 & 2 \\ a & b & a \end{pmatrix}$$

$$V_7 = \begin{pmatrix} 1 & 1 & 2 \\ a & b & b \end{pmatrix}.$$

Algorithm 1 *Gen_polyominoes*(s, m, k, x)

> $V[1, s-m+1] = k$; $V[2, s-m+1] = x$;
> **if** $m = 1$ **then**
>   output V
> **else**
>   **if** $x = a$ **then**
>   *Gen_a*(m, k)
>   **else if** x=b **then**
>   *Gen_b*(m, k)
>   **else if** x=g **then**
>   *Gen_g*(m, k)
>   **else**
>   *Gen_r*(m, k)
>   **end if**
> **end if**

end

Under the assumption that $1 \leq m \leq s$, the call *Gen_polyominoes*(s, m, k, x), generates all the paths of length m starting from a node with label $(k)_x$ located at level $s-m$ in the generating tree of Ω. More precisely, *Gen_polyominoes*(s, m, k, x) calls the procedure *Gen_x*(m, k), which, on its turn, recursively calls *Gen_polyominoes*$(s, m-1, k', x')$ for each $(k')_{x'}$ being produced by $(k)_x$ according to (8).
For brevity we describe a pseudo-code only for the Algorithm *Gen_polyominoes*, and for the Procedure *Gen_b*. The three other procedures *Gen_a*, *Gen_g*, and *Gen_r* are analogous to *Gen_b*, and follow the productions in the rule (8).

algorithmProcedure

Algorithm 2 *Gen_b*(m, k)

> **for** $j = 1$ to $k-2$ **do**
>   **for** $j' = 1$ to $k-j-1$ **do**
>   *Gen_polyominoes*$(s, m-1, j, g)$
>   **end for**
> **end for**
> **for** $j = 1$ to $k-1$ **do**
>   *Gen_polyominoes*$(s, m-1, j, r)$; *Gen_polyominoes*$(s, m-1, j, r)$
> **end for**
> *Gen_polyominoes*$(s, m-1, k, b)$
> *Gen_polyominoes*$(s, m-1, k+1, a)$
> *Gen_polyominoes*$(s, m-1, k+1, b)$

end

Since the root of the associated generating tree is $(1)_a$, we have that:

Lemma 4.1 *The call Gen_polyominoes*$(n+1, n+1, 1, a)$ *generates all the* f_n *polyominoes of semi-perimeter* $n+2$.

Now, we can prove that:

Proposition 4.1 *The algorithm for the generation of convex polyominoes runs in constant amortized time.*

Proof. Let Op_n denote the number of operations required by the algorithm to generate all the convex polyominoes with semi-perimeter $n+2$.
The algorithm performs a prefix traversal in the first $n+1$ levels of the generating tree, and the visit of each node requires a constant number α of operations ($\alpha \leq 6$). Since the algorithm is recursive, we easily obtain the recurrence relation:

$$Op_n = Op_{n-1} + \alpha f_n,$$

where, as usual, f_n is the expression defined in equation (1). Our statement is proved showing by induction that:

$$\frac{Op_n}{f_n} < 2\alpha.$$

The above inequality clearly holds for $n = 0$. Let us consider the inductive step:

$$\frac{Op_n}{f_n} = \frac{Op_{n-1}}{f_{n-1}} \frac{f_{n-1}}{f_n} + \alpha.$$

Since it is not difficult to prove that $2f_{n-1} < f_n$, by inductive hypothesis we get:

$$\frac{Op_n}{f_n} < 2\alpha \, \frac{1}{2} \; + \; \alpha = 2\alpha,$$

which means that, using our algorithm, to generate convex polyominoes with fixed size requires an average amount of computation that is bounded by a constant. □

4.2 Exhaustive generation of column-convex polyominoes

The ECO operator ϑ_1, defined in Section 2.2, describes a recursive construction for the class of column-convex polyominoes, and the associated succession rule is Δ, in (11).

The same ideas of Section 4.1 can thus be applied in order to determine an algorithm for the exhaustive generation of this class. What remains to prove is that also in this case the generation algorithm runs in constant amortized time.

This result can be achieved in a simple way, repeating the same steps of Proposition 4.1. In particular, using (5) we easily obtain that $3g_{n-1} < g_n$, with $n > 1$, and then by induction we prove that:

$$\frac{Op_n}{g_n} < \frac{3}{2}\alpha,$$

where, as usual, Op_n denotes the number of operations required by the algorithm to generate all the column-convex polyominoes with semi-perimeter $n + 2$.

5 Supplementary remarks

The enumeration of convex or of column-convex polyominoes is often classed as "difficult". Certainly, previously published solutions tend to skirt the issue of why the generating functions for these classes of polyominoes turn out to be algebraic. In contrast, the solution presented here proceeds by means of sequence of simple steps resulting in a system of equations for the generating functions that, *ipso facto*, indicates that they are algebraic. Perhaps an analogy can be made here with the comparable way in which the *transfer matrix method* works to somewhat the same effect.

As a bonus, our method delivers additional information about convex polyominoes through their interpretation as terms of an algebraic formal power series. Indeed, since the series $A(x)$ appearing in the system (31) represents in this way the class of convex polyominoes, the other series in this system also represent, in effect, special sub-classes of convex-polyominoes. To give just one illustration, the series $(1)_a B(x)$ represents the class of of convex polyominoes having at least two cells in the first column. Thus, the system (31) can be interpreted as an *object grammar* decomposition for convex polyominoes and consequently is a store of combinatorial meaning, again not unlike a *transfer matrix* (for object grammars, we refer to [16]).

We feel that this approach is promising and intend to try extending it, especially in other cases where an ECO construction is already known or can be determined, although we do not envisage the method as being always so appropriate.

We conclude by mentioning some further observations:

1. Our approach relying on the ECO method conjoined with formal power series can be deployed in the study of several other statistics relating to convex and column-convex polyominoes. By way of example, the reader might care to verify that, considering the productions in (8), but using an axiom different from that of the rule Ω as described below, yields the enumeration of other classes of convex polyominoes, indexed as usual by semi-perimeter:

   (a) if the axiom is chosen to be $(k)_a$, $k \geq 1$, the rule gives the enumeration of the class of convex polyominoes having at least k cells in the first column. With $k = 1$ we get Ω;

   (b) if the axiom is chosen to be $(k)_r$, then the rule defines the class of directed-convex polyominoes having at least k cells in the first column, $k \geq 1$;

   (c) if the axiom is chosen to be $(k)_g$, then the rule defines the class of *stack polyominoes* or *stacks* having exactly k cells in the first column, $k \geq 1$ (for a further, separate account of stacks, see [27]).

   Not surprisingly, the computation of the generating function for these rules is similar to that for the rule Ω.

2. Combinatorial structures having an ECO method of construction, of which a varied selection appear in [4, 19], can be exhaustively generated using an approach similar to that in Section 4. In addition, as shown in [15], a large class of regular languages can also be generated in a *quasi*-automatic way by the ECO method, and each language in this class then admits a similar algorithm. A general algorithm that uses ECO systems to derive a generating algorithm has been recently developed in [2], where some partial criteria to establish whether a given ECO system leads to an algorithm with the CAT property are also presented. But it remains a challenge to delineate necessary and sufficient conditions for an ECO system to yield a CAT algorithm. For that matter, it would be interesting just to have examples of ECO systems that fail in this regard.

Acknowledgements. We dedicate this work to Renzo Pinzani, as a token of our appreciation for the inspiring example he has given us, both as a teacher, when we were still students, and as a fellow researcher. The ECO method on which we have drawn extensively throughout this paper is, of course, very much a product of Professor Pinzani's *school* at the University of Florence, of which it has been our privilege to be members.

Because of the central place Alberto Del Lungo also occupied in this research *milieu*, his sudden and unexpected death during the writing of this paper was not only a great shock and sorrow to the surviving authors, but brought them a sense of lost promise for future collaborative work. Nevertheless, they hope that this last paper on which Professor Del Lungo worked will serve as a substantial memorial to him.

References

[1] D. Avis, K. Fukuda, Reverse search for enumeration, *Discrete Appl. Math.* 65 (1996) 21-46.

[2] S. Bacchelli, E. Barcucci, E.Grazzini, E. Pergola, Exhaustive generation of combinatorial objects by ECO, *Acta Informatica* 40 (2004) 585-602.

[3] C. Banderier, M. Bousquet-Mélou, A. Denise, P. Flajolet, D. Gardy and D. Gouyou-Beauchamps, Generating functions for generating trees, *Discrete Math.* 246(1-3)(2002) 29-55.

[4] E. Barcucci, A. Del Lungo, E. Pergola, R. Pinzani, ECO: a methodology for the Enumeration of Combinatorial Objects, *J. of Diff. Eq. and App.* 5 (1999) 435-490.

[5] D. Beauquier, M. Nivat, On translating one polyomino to tile the plane, *Disc. Comput. Geom.* 6 (1991) 575–592.

[6] M. Bousquet-Mèlou, A method for the enumeration of various classes of column-convex polygons, *Discrete Math.* 154 (1996) 1–25.

[7] M. Bousquet-Mèlou, A. J. Guttmann, Enumeration of three dimensional convex polygons, *Ann. of Comb.* 1 (1997) 27–53.

[8] R. Brak, A. J. Guttmann, I. G. Enting, Exact solution of the row-convex polygon perimeter generating function, *J. Phys.* A 23 (1990) L2319–L2326.

[9] S. Chaiken, D. J. Kleitman, M. Saks and J. Shearer, Covering regions by rectangles, *SIAM J. Discr. and Alg. Meth.* 2 (1981) 394–410.

[10] S. J. Chang, K. Y. Lin, Rigorous results for the number of convex polygons on the square and honeycomb lattices, *J. Phys. A: Math. Gen.* 21 (1988) 2635–2642.

[11] F. R. K. Chung, R. L. Graham, V. E. Hoggatt, M. Kleimann, The number of Baxter permutations, *J. Comb. Th. A* 24 (1978) 382–394.

[12] J. H. Conway, J. C. Lagarias, Tiling with polyominoes and combinatorial group theory, *J. Comb. Th. A*, 53 (1990) 183–208.

[13] M. Delest, Generating functions for column-convex polyominoes, *J. Comb. Th. Ser.* A, 48 (1988) 12-31.

[14] M. Delest, X. Viennot, Algebraic languages and polyominoes enumeration, *Theor. Comp. Sci.*, 34 (1984) 169–206.

[15] E. Duchi, A. Frosini, S. Rinaldi, R. Pinzani, A note on rational succession rules, *J. Int. Seq.* 6 (2003) Article 03.1.7.

[16] I. Dutour, J.M. Fédou, Object grammars and random generation, *Disc. Math. and Theor. Comp. Sci.*, 2 (1998) 47–61.

[17] J. M. Fedou, C. Garcia, Algebraic succession rules, *Proc. of 14th FPSAC* (2002) n.27.

[18] L. Ferrari, E. Pergola, R. Pinzani, S. Rinaldi, An algebraic characterization of the set of succession rules, *Theor. Comp. Sci.*, 281(1-2) (2002) 351–367.

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

[20] S. Feretić, A new way of counting the column-convex polyominoes by perimeter, *Discrete Math.*, 180 (1998) 173–184.

[21] I. Gessel, On the number of convex polyominoes, *Annales des Sciences Mathemathiques du Quebec*, 24 (2000) 63–66.

[22] S. W. Golomb, Checker boards and polyominoes, *Amer. Math. Monthly*, vol. 61, n. 10 (1954) 675–682.

[23] J.M. Hammersley, Percolation processes II: the connective constant, *Proc. Cambridge. Philos. Soc.*, 53 (1957) 642–645.

[24] D. Kim, The number of convex polyominoes with given perimeter, *Discrete Math.* 70 (1988) 47–51.

[25] B. D. McKay, Isomorph-free exhaustive generation, *J. Algorithms* 26 (1998) 306–324.

[26] D. H. Redelmeier, Counting polyominoes: yet another attack, *Discrete Math.*, 36 (1981) 191–203.

[27] S. Rinaldi and D. G. Rogers, How the odd terms in the Fibonacci sequence stack up, *Math. Gaz.*, to appear.

[28] N. J. A. Sloane, The On-Line Encyclopedia of Integer Sequences, published electronically at *http://www.research.att.com/∼njas/sequences/.*

[29] H. N. V. Temperley, Combinatorial problems suggested by the statistical mechanics of domains and of rubber-like molecules, *Phys. review* 2 103 (1956) 1–16.

[30] J. West, Generating trees and the Catalan and Schröder numbers, *Discrete Math.* 146 (1996) 247–262.

[31] J. West, Generating trees and fordibben subsequences, *Discrete Math.* 157 (1996) 363–374.

[32] W.-J. Woan, L. W. Shapiro, D. G. Rogers, The Catalan numbers, the Lesbesgue integral, and 4^{n-2}, *Amer. Math. Monthly* 104 (1997) 926–931.