Combinatorics, information vizualisation and algebraic languages

Maylis DELEST

Université Bordeaux 1, France
maylis@labri.u-bordeaux.fr

1. Introduction

Let Ω be a class of combinatorial objects. Let us suppose that they are enumerated by the integer a_n according to the value n of some parameter p. Let $f(t) = \sum_{n=0} a_n t^n$ be the corresponding generating function. One of the main problem addressed by combinatorics is founding $f(t)$ and its properties knowing a recurrence on a_n or even the sequence of the first value a_0, a_1, a_2, ... Many books are devoted to this field [Ri, Wi]. With computer algebra systems, new techniques have been set up for getting results [Ma, Mu]. Internet network users may ask on-line information on a sequence of values [S]. This last service is an encyclopedia of integers sequences but also it gives useful references to objects that are counted by the sequences. Enumerative combinatorics is focussed on getting more inside the formula using bijection with object classes. Let us give an old trite example due to Euler in order to enlight what we call *getting inside formula*. Let s_n defined by

$$s_n = \sum_{i=0}^{n} (2i+1)$$

Of course, we have $s_n = (n+1)^2$. This result can be obtained by algebra but also explained by a geometrical construction. For each i, the value $(2i+1)$ is represented by a hook of $(2i+1)$ cells in the plane $\mathbb{N} \times \mathbb{N}$. Then, the s_n value is constructed by putting the hooks upon each other. See Figure 1.

In this paper, we will focus on methods intensively studied by the *Combinatorics Bordeaux School* and some of their applications. After some definitions and notations, we describe in section 3 the DSV methodology and in section 4 two extensions that are object grammars and Q-grammars. At the end, we show applications of these techniques to information visualization.

2. Definitions and notations

This section summarizes briefly the notions needed for understanding this paper. A more complete background can be acquired from [Be, AU]. Let X be a nonempty set called alphabet. The elements of X are called letters. A word is a finite sequence of letters from X. The empty word is usually denoted by ϵ. Let u and v be two words on X, $u = u_1...u_p$ and $v = v_1....v_q$. We define the concatenation of two words to be $uv = u_1...u_p v_1....v_q$. We

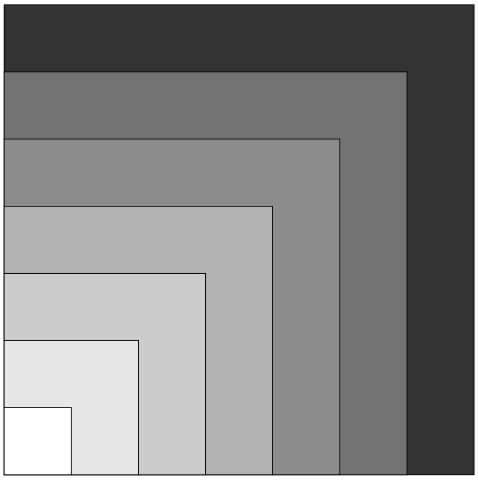


Figure 1: Euler proof for $n = 6$, $s_n = 49$.

denote by X^* the free monoid generated by X, that is, the set of all words on X endowed with the operation of concatenation. The number of occurrences of the letter x in the word u is denoted by $|u|_x$. The number of letters of a word w is called length of w and is denoted by $|w|$. A language is a subset of X^*. To every language $\mathcal{L}$, one can associate a noncommutative formal power series

$$L = \sum_{w \in \mathcal{L}} w$$

that is an element of the algebra $\mathbb{Z} \ll X \gg$ of noncommutative formal power series with variables in X and coefficients in $\mathbb{Z}$.

Definition

An algebraic grammar is a 4-uple $G =< N, X, P, s >$ such that N and X are two disjoint alphabets called, respectively, the non-terminal and the terminal alphabet, s is an element of N called axiom, and P is a set of pairs (α, β) with $\alpha \in N$ and $\beta \in (N \cup X)^*$ called production rules and denoted by $\alpha \to \beta$.

Let α be in N and u in $(N \cup X)^*$, $u = u_1 \alpha u_2$. A derivation in G is a rewriting of u as $v = u_1 \beta u_2$ with $\alpha \to \beta$. This will be denoted by $u \to v$. We say that a word w is deriving from a non-terminal symbol α in G if there exists a sequence of derivations which rewrites α as w. This will be denoted by $\alpha \xrightarrow{*} w$. The set $L(G)$ of words generated by s is called the algebraic language generated by G. In general, there may exist several grammars for a given algebraic language.

Example

The main example in combinatorics is the Dyck language, not because of its complexity, but because of the frequency with which it occurs in different settings. It encodes numerous and diverse structures such as trees, paths, polyominos, ... Its words are generated by the grammar G_1 given by

$$N = \{D\}, X = \{x, y\}, s = D,$$

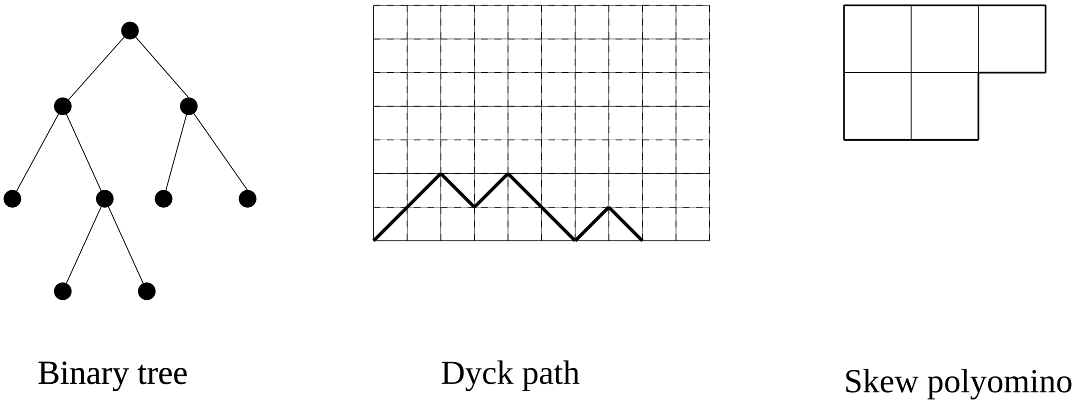


Figure 2: Combinatorial objects encoded by the word xxyxyyxy.

and the production rules

$$D \to xDyD, D \to \epsilon$$

This example give rise to unambiguous algebraic grammars, that is, algebraic grammars in which every word is obtained only once from the axiom using the production rules in a left-right derivation that is deriving first the leftmost terminal. In such cases, the formal power series associated to the language verifies equations which follow directly from the production rules. In our example,

$$D = xDyD + \epsilon.$$

In the following, all the grammars are considered to be unambigous.

3. DSV methodology

This methodology stems from an idea of M.P. Schützenberger from 1959 [S62, S63]. This method is now known as the DSV-methodology, following M.P. Schützenberger's wish expressed to Viennot [Vi91].

Let X be an alphabet, $X = \{x_1, ..., x_k\}$. The commutative image of a series produces, from a noncommutative formal power series, a commutative one, called an enumerative series of the language $\mathcal{L}$. This is defined by:

$$\chi_0(\mathcal{L}) = \sum_{i_1, i_2, \ldots, i_k \in \mathbb{N}^k} n_{i_1, i_2, \ldots, i_k} x_1^{i_1} x_2^{i_2} \ldots x_k^{i_k}$$

such that $n_{i_1,i_2,\ldots,i_k}$ is the number of words w in $\mathcal{L}$ such that $|w|_{x_j} = i_j$ for each j in $[1..k]$. In this way, we obtain an application from the boolean semi-ring $\mathbb{B} \ll X \gg$ to the semi-ring $\mathbb{N} \ll X \gg$ of commutative formal power series with variables in X. We will often denote by L the series $\chi_0(\mathcal{L})$. The application χ_0 is not a morphism but the following theorem hold.

Theorem 1 *The image of an algebraic language under χ_0 is an algebraic series which is one of the components of the solution of the system of equations obtained via χ_0 from an unambiguous grammar of the language.*

Example

The computation on the Dyck language classically leads to the equation :

$$D(x,y) = xyD(x,y) + 1$$

An elementary computation shows that

$$D(x,y) = \frac{1 - \sqrt{1-4xy}}{2x^2},$$

from which we deduce easily that the number of Dyck words of length 2n is the Catalan number

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

Numerous results, in several areas, were obtained by this method : polyominos [DV, De88] tRNA structures [VV]. Some overviews can be found in [De91, Vi92].

Remark

Some algebraic languages cannot be associated to unambiguous grammars. Flajolet [Fl] has shown that their generating series are related to transcendental series.

So, the first step in the Schützenberger methodology, namely the encoding, requires particular insight: one must find a bijection between the objects and an algebraic language. We remark that, frequently, the language which is obtained in the bijection process turns out to be closely related to the Dyck language. We will see in section 4 an explanation of this fact. Thus, n the following, we describe bijections linked directly to Dyck words. Let us consider the set B of binary trees. If b is in B, then it admits the following recursive description: either $b = (root(b), L(b), R(b))$ where $root(b)$ is an internal node called root and $L(b)$ ($R(b)$, resp.) is the left (right, resp.) tree, or b is a single point called leaf. To encode a tree by a Dyck word, traverse the tree in left first depth-first order (or prefix order, that is, visiting first the root, then the left subtree, then the right subtree). During the traversal, write x at each internal node and y at each leaf, except the last one. This is the classical bijection between binary trees and Dyck words. One deduces the following well-known result.

Theorem 2 *The number of binary trees having n+1 leaves is the Catalan number C_n.*

Let us now consider the set of paths in $\mathbb{N}\times\mathbb{N}$ which are sequences of points $(s_0, s_1, \ldots, s_n)$. The pairs (s_i, s_{i+1}) are called elementary steps. They are North-East (South- East, resp.) if $s_i = (k, k')$ and $s_{i+1} = (k+1, k'+1)$ (resp. $s_{i+1} = (k+1, k'-1)$). The height of the step s_i is k'. These paths are clearly in bijective correspondence with Dyck words: simply follow the path from s_0 to s_n, encoding each NE step by x, and each SE step by y. The word obtained in this manner is a Dyck word if and only if $s_0 = (0,0)$ and $s_n = (\lfloor n/2 \rfloor, 0)$. The corresponding path is frequently referred to a Dyck path. Another subject where the Schützenberger methodology gives good results is that of polyomino enumeration. For surveys, we refer the interested reader to [De91, Gu, Vi92]. A polyomino can be described as a finite connected union of cells (unit squares) in the plane $\mathbb{N}\times\mathbb{N}$, without cut points. A column (row, resp.) of a polyomino is the intersection of the polyomino with an infinite vertical (horizontal, resp.) strip of cells. A polyomino is column-convex (row-convex, resp.) if every column (row, resp.) is connected. A skew polyomino is both row- and column-convex and for each one of its columns there is

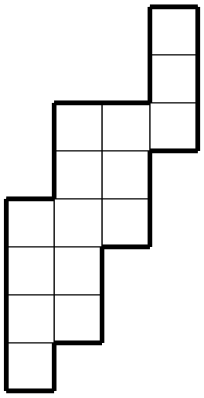


Figure 3: Euler proof for $n = 6$, $s_n = 49$.

- no column on its right with a cell lower than its lowest cell,
- no column on its left with a cell higher than its highest cell.

An analysis of these constraints leads to an alternate definition of a skew polyomino, as a pair of integer sequences $(a_1, \ldots, a_n)$ and $(b_1, \ldots, b_{n-1})$, where a_i is the number of cells belonging to the ith column and $b_i + 1$ is the number of adjacent cells from columns i and $i + 1$. On figure 3, is displayed a skew polyomino and the two sequences a and b. These two sequences can be viewed as the heights of the peaks (step North-East followed by a step South-East) and the heights of the troughs (step South-East followed by a step North-East) in a Dyck path. So encoding a skew polyomino by a Dyck word is straightforward and it is easy to deduce the next result.

Theorem 3 *The number of skew polyominos whose perimeter equals $2n + 2$ is the Catalan number C_n.*

This result was already known a long time ago. The bijection from the Schützenberger methodology merely explains combinatorially the link between polyominos and Catalan numbers. The first new result [DV] in enumerative combinatorics obtained by this methodology pertains to convex polyominos.

Theorem 4 *The number p_{2n} of convex polyominos having a perimeter $2n + 8$ is*

$$p_{2n} = (2n + 11)4^n - 4(2n + 1) \binom{2n}{n}$$

This result was first proved with bijection with languages constructed from Dyck one and heavy computation. A totally bijective proof was given by Mireille Bousquet-Mélou [Bo94].

4. Extension of DSV-methodology

4.1. Q-grammars

The study of compilers in computer science shows that the semantic attribute method described by Irons [Ir61, Ir63] and then by Knuth [Kn] allows the translation of words from an

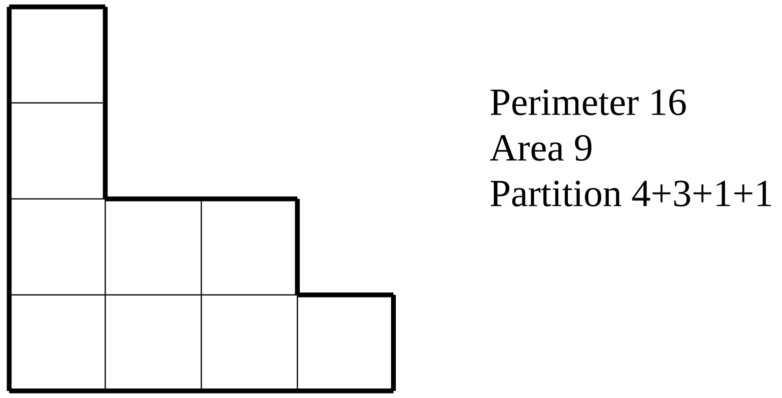


Figure 4: A Ferrer diagram.

algebraic language. Most of the resulting translations, however, are not algebraic languages. In the context of enumerative combinatorics, the same set of objects may lead to an algebraic generating function if counted according to a certain parameter, and to a non-algebraic one if counted according to another. For example, the generating function for Ferrers diagrams (that is representation of partition of an integer, see figure 4) is algebraic according to the perimeter of the diagram

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

and not algebraic according to the number of cells

$$f(q) = \prod_{i=1}^{\infty} \frac{1}{1-q^i}.$$

The interest presented by the attribute method lies in the fact that translation is defined locally, on each production rule of the grammar. Formally, in the combinatorics background, we have the

Definition

Let $G = (X, N, P, S)$ be a grammar. for each $U \in N$, an attribute family defined on G is given by a finite set T_U of attributes.

- Each attribute $\tau \in T_U$ has a domain D_τ; the cartesian product $\prod_{\tau \in T_U} D_\tau$ is denoted by $\mathcal{D}_U$;
- For each attribute $\tau \in T_U$ and for each derivation in P of U, $R : U \to w_0 U_1 \ldots U_k w_k$, a computation rule is defined $f_{\tau,R}$, that is a function from $\mathcal{D}_{U_1} \times \ldots \times \mathcal{D}_{U_k}$ into D_τ.

$\left(G, (T_U)_{U \in N}\right)$ is called an attribute grammar

For each word $w \in L(G)$, and each attribute τ, the function describe the recursive computation of $\tau(w)$.

It can be shown (see [17], [SFCA]) that if the attribute system is well defined (in a sense that we will not explain here), then a system of q-equations can be obtained directly from the q-grammarattribute grammar. Adding attributes to a grammar introduces non-algebraic substitutions in the commutative equations. Here, we just give a trite example.

Example

The language coding Ferrers diagrams is the language encoding their profile by means of words of the form $w = aub$ written on the alphabet $\{a, b\}$. A grammar for this language is

$$G =< \{S, L\}, \{a, b\}, \{S \to aLb, L \to aL, L \to bL, L \to \epsilon\}, S >$$

The attribute grammar $(G, \tau$ defined below computes the number of cells based on this encoding:

$$S \to aLb, \tau(S) = q^{|\tau(L)|_a + |\tau(L)|_b + 1} ab\tau(L),$$

$$L \to aL, \tau(L) = q^{|\tau(L)|_b} a\tau(L),$$

$$L \to bL, \tau(L) = b\tau(L),$$

$$L \to \epsilon, \tau(L) = 1.$$

It is easy to show that the system of q-equations is

$$S(a, b; q) = qaL(aq, bq; q)b,$$

$$L(a, b; q) = aL(a, bq; q) + bL(a, b; q) + 1,$$

from which one can deduce the well-known generating function

$$s(a, b; q) = \sum_{n=0}^{\infty} \frac{a^n q^{n+1}}{(1 - qb)(1 - q^2 b) \ldots (1 - q^{n+1} b)}.$$

Systems of q-equations can be obtained by this method, but solving them remains challenging even in cases when they give very nice results (see [DF93]). We must point out a general solution for q-equation given by M. Bousquet-Mlou in [Bo96].

4.2. Object grammars

Another extension of the DSV-methodology is object grammars, due to Dutour and Fdou [DF97]. They describe a Schützenberger method without word that is based only on the unambigous recursive decomposition of the combinatorial objects. Other methods have a similar approach see [FSZ]. Let us give a definition, then we will describe some applications. Let $\mathcal{E}$ a set of combinatorial objects.

Definition

Let $\{E_i\}_{i=1,k}$ and E be subsets of $\mathcal{E}$. An object operation is an application from $E_1 \times E_2 \times \ldots \times Ek$ in E.

Definition

An object grammar is a 4-uple $G =< \mathcal{F}, \mathcal{T}, P, \int >$ such that

- $\mathcal{F}$ is a set of subset of $\mathcal{E}$,
- $\mathcal{T}$ is a finite set of terminal objects in $\mathcal{E}$,

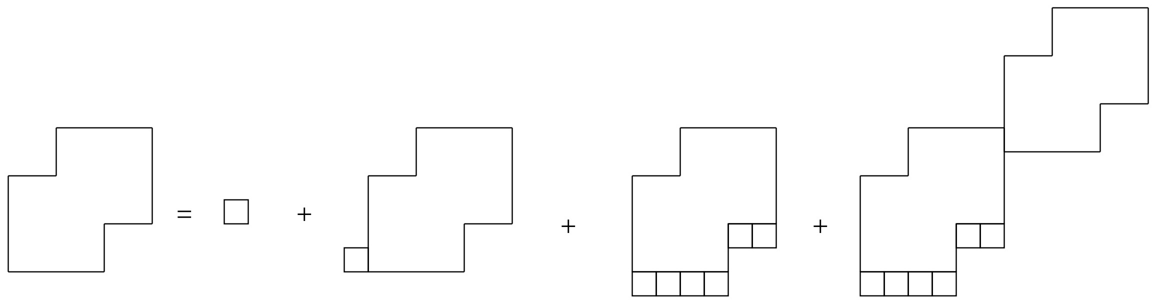


Figure 5: An object grammar for skew polyominoes.

- P is a set of object operations defined on k-uples of $\mathcal{F}$ with value in $\mathcal{F}$,
- $\int$ is called axiom.

Clearly if $\mathcal{T}$ is an alphabet and $\mathcal{E} = \mathcal{T}^*$ then an algebraic language can be defined by an object grammars. As for algebraic language, one can define the derivation of an object from $\int$ in the object grammar G.

Example

Let us come back to skew polyominoes. Let E be the set of polyominoes, $\mathcal{S}$ be the set of skew polyominoes. We define :

- $\mathcal{F} = \{\mathcal{S}\}$,
- $\mathcal{T} = \{\Box\}$ that is the polyomino with only one cell,
- $\int = \mathcal{S}$.

Let O_1 and O_2 be skew polyominoes, then the objects operations are the followings :

- ϕ_1 consists in adding to O_1 a column with one cell on its left in order to get a new skew polyomino,
- ϕ_2 consists in adding one cell to each column of O_1
- ϕ_3 consists in gluing two polyominoes O_1 and O_2 by the rightmost upper cell of O_1 and the leftmost downer cell of O_2.

Clearly we have the recursive equation :

$$\mathcal{S} = \Box + \phi_1(\mathcal{S}) + \phi_2(\mathcal{S}) + \phi_3(\phi_2(\mathcal{S}), \mathcal{S})$$

that is pictured in figure 5)

In enumerative combinatorics, Catalan numbers and Dyck words are involved in a lot of construction. One central result in objects grammars enlights this fact.

let $G =< \mathcal{F}, \mathcal{T}, P, \int >$ be an object grammar such that $\mathcal{F} = \mathcal{D}$. Dutour anf Fédou associate a caracteristic polynomial $g(x)$ of G. Let us define it for a one dimensional system. It is obtained by substituing in the right part of the rule associated to $\mathcal{D}$ each occurence of $\mathcal{D}$ by x and each terminal object by 1, forgetting the object operations.

Example

From the previous equation, we get the $g(x) = 1 + 2x + x^2$

Using the notion of substitution and constructing the solution, they proved the following theorem :

Theorem 5 *Two one dimensional grammars of degree almost two are isomorphic by the substitution process.*

As a consequence, for a large class of combinatorial objects, bijections can be constructed from the Dyck language. Nice example are given in [DF97] Of course, the notion of polynomial can be extended to system having higher degree than one. Moreover, they give an efficient tool for the random generation of objects. The Maple package is available at

http://dept-info.labri.u-bordeaux.fr/ dutour/QALGO/.

5. Application to information visualization

In this section, we describe the software Latour [HDM] developped by CWI and LaBRI. This software deals with tree visualization. The problem of displaying and interacting with large set of information can be abstracted to the same problem for graphs. Latour is devoted to the special case of tree. A lot of software try to have a very good drawing of the structure, [BETT]. The scale of information visualization raises up structures having frequently several thousands of nodes. Insteed of trying to solve the complete problem, the Latours's approach is to use enumerative combinatorics in order to construct tools for exploring or folding parts of the structure.

In figure 6), a tree is shown with Latour menus. The main goal in Latour is to construct measures on the tree that help the user in watching informations. Here, we want to enlight only two measures : guiding the user in a zoom function and folding automatically subtrees that are too big or too small .

5.1. Strahler numbers as a user guide

Strahler numbers are very classical numbers in a lot of fields as biology, computer science [Vi92]. They were defined by the geographs Norton and Strahler in order to give a mathematical definition for fluvial bassin. The definition for Strahler numbers on binary trees can be done as follow.

Definition

Let $b = (root(b), L(b), R(b))$ be a binary tree. To each node $v \in b$, the Strahler number $S(v)$ of v is

- if v is a leaf then $S(v) = 1$,
- if $S(root(L(b))) = S(root(R(b)))$ then $S(v) = S(root(L(b))) + 1$,
- else $S(v) = \max(S(root(L(b))), S(root(R(b))))$.

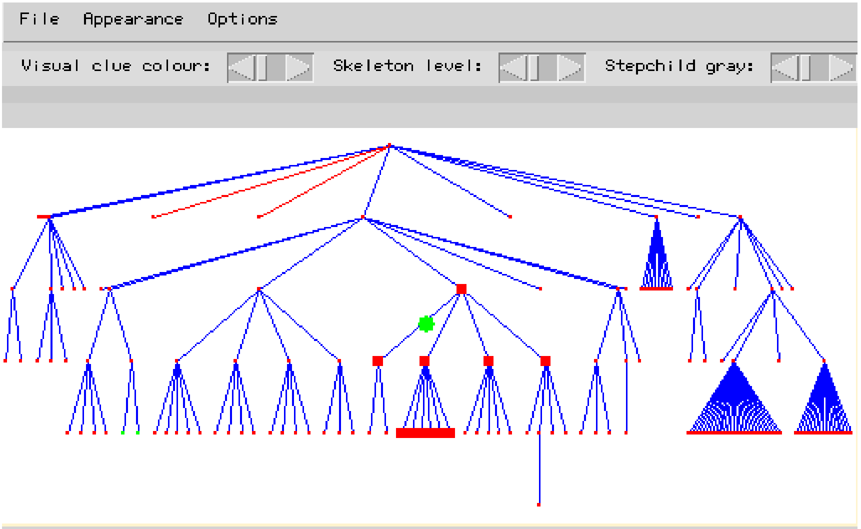


Figure 6: A view of the Latour software.

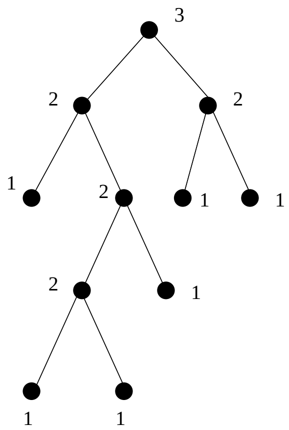


Figure 7: Strahler number computation for binary trees.

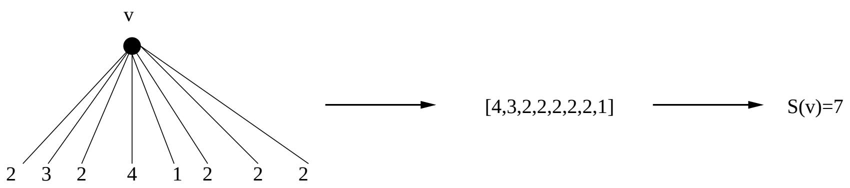


Figure 8: Strahler number computation for plane trees.

The Strahler number of the tree is $S(root(b))$.

An example is displayed on figure 7. Many extension of Strahler numbers can be set for plane trees. One of them, due to Fédou, is meaningfull according to computer science definition that is minimum number of registers for computing an arithmetical expression [DDDF]. In this case the Strahler numbers are given rather by an algorithm than by a formula. Roughly, the value of the leaves are 1. Suppose that a vertex v of the tree has sons $v_1, v_2, ...v_k$ then rank the $v_{i i=1..k}$ in decreasing order, it gives a list of value $u(i)_{i=1..k}$. Then do

```
s:=u(1);
free:=u(1)-1;
for i from 2 to k do
  if free<u(i) then s:=s+1
               else free:=free-1
  fi
od;
```

Then the variable s contains the value of $S(v)$ (see figure 8).

A part of the mathematical sudy of this parameter on trees, coloring edges according to this parameter in a tree visualization software guide the user during a zoom. It shows him which relative importance has the zoom window in the whole tree. We have experimented this technic in several field (data structures for compilers, file hierarchical systems, ...). In figure 9(resp. figure 10) , we give two views of the same tree with zoom without (rep. with) Strahler measure.

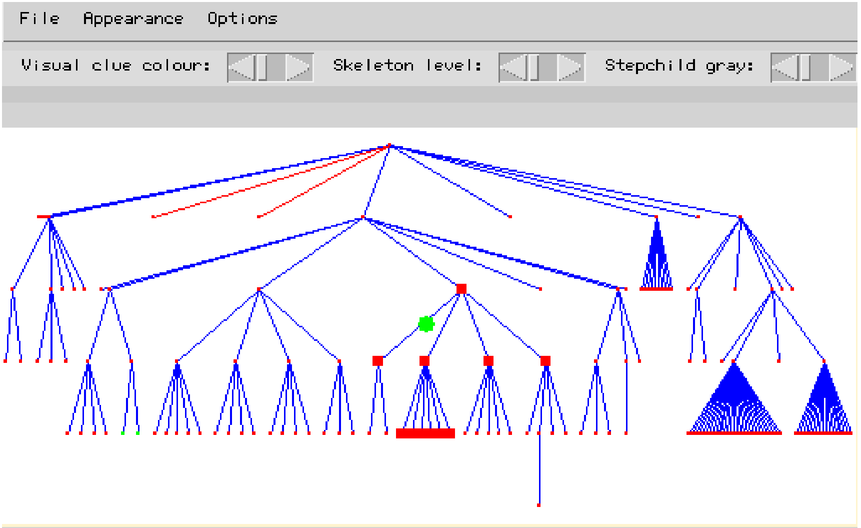
5.2. Folding trees using leaves numbers

The number of leaves of a plane tree is a very classical parameter. We have the well-known result

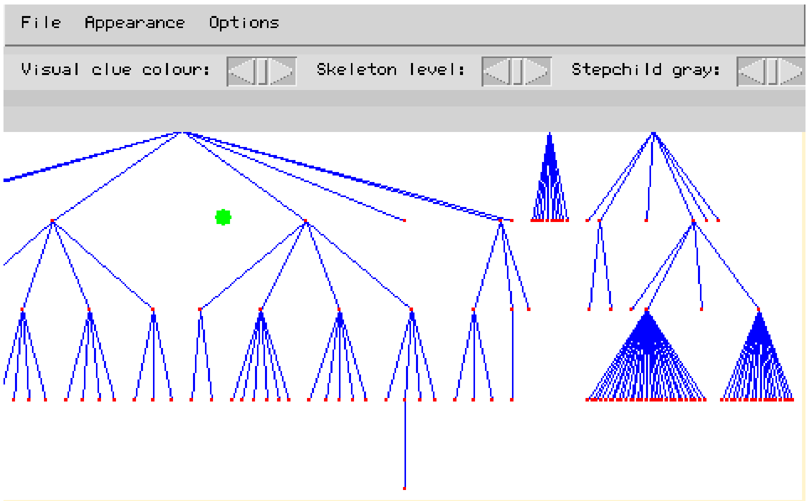
Theorem 6 *The number of tree B having n nodes and k leaves is*

$$C_{n,k} = \frac{1}{n-1} \binom{n-1}{k} \binom{n-1}{k-1}.$$

Let f_n be the random variable number of leaves in a tree having n nodes. Then, as soon as n is greater than 10, f_n as a normal distribution with mean $n/2$ and standard deviation $\sqrt{(\frac{n}{8})}$. This approximation can be deduces based on a much more general (and highly non trivial) theorem described in a paper of M. Dmrota [Dr].

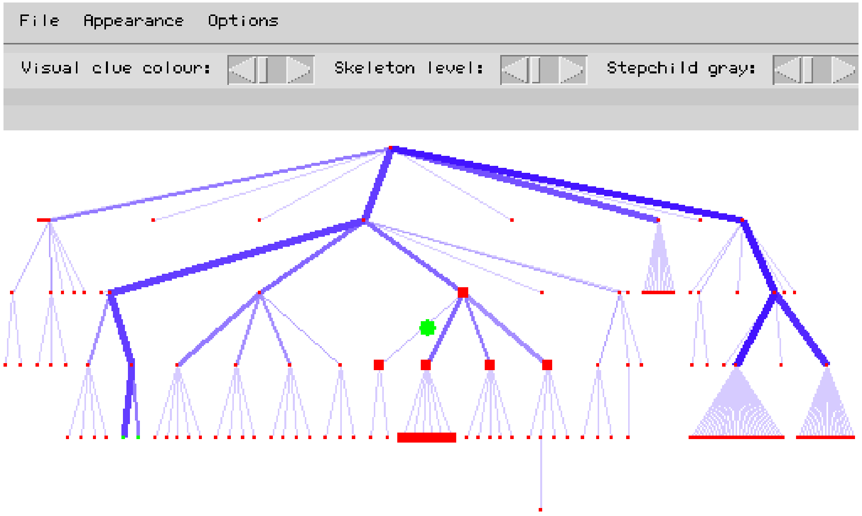


A

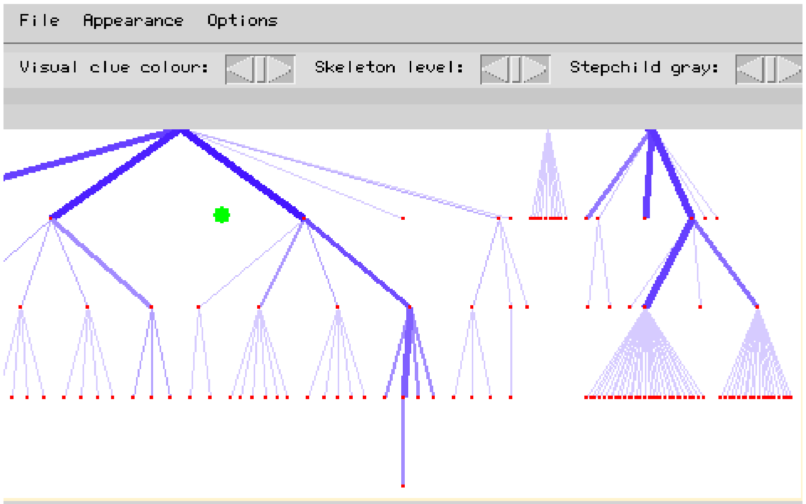


B

Figure 9: Views from latour : (A) Simple view, (B) Simple zoom view.



A



B

Figure 10: Views from latour : (A) Strahler View, (B) Strahler zoom.

In Latour, we use this this distribution for folding automatically subtrees that are "unusually" large or small in regard of their number of leaves. We must point out, that, at this stage, we don't take account of the number of leaves of the full tree. This can change drastically the probability law. In the future, the fold tool will offer to the user to take account of the number of leaves in the general tree and also two others features :

- the maximum degree of the nodes,
- the maximal length of paths such that each node has only one son.

Anyway, users agree on this tool. In figure 11, we show the folding effect. The Latour software is available at http://www.cwi.nl/InfoVisu.

References

References

[AU] A. Aho, J. Ullmann, Introduction to automata theory, languages, and computation, Addison Wesley, 1979.

[BETT] G. Di Battista, P. Eades, R. Tamssia & L.G. Tollis, Algorithms for drawing graphs an annotated bibliography, *Computational Geometry : Theory and Applications*, 4 **5** 1994, 235-28.

[Be] J. Berstel, Transductions and context-free languages, Teubner Verlagsgesellschaft, 1979.

[Bo94] M. Bousquet-Mélou, Codage des polyominos convexes et quations pour l'numration suivant l'aire, *Discrete Appl. Math.* 48 (1994), 21-43.

[Bo96] M. Bousquet-Mélou, A method for the enumeration of various classes of column-convex polygons , *Discrete Math.* 154 (1996), 1-25.

[DDDF] M. Delest, J.P. Domenger, P. Duchon & J.M. Fédou, Strahler numbers for plane trees, to appear.

[DF93] M. Delest, J.M. Fdou, Enumeration of skew Ferrers diagrams, *Discrete Maths*, 112 (1993), 65-79.

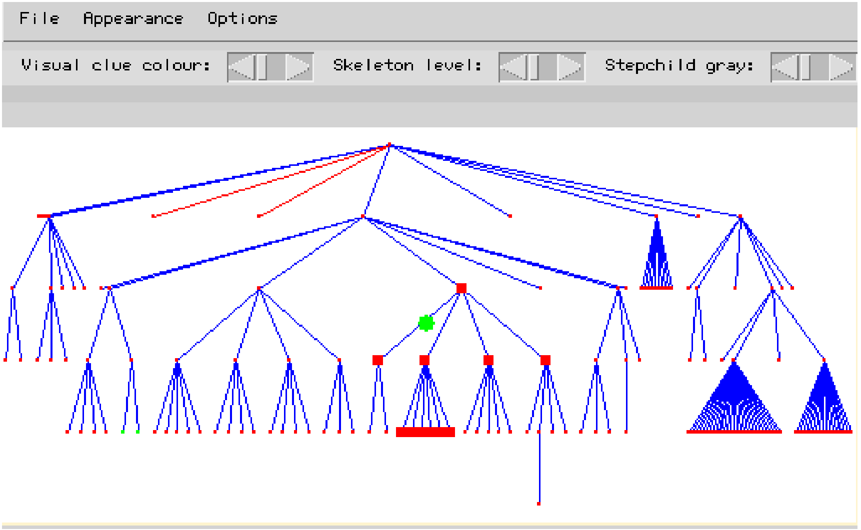
[Dr] Asymptotic distributions and a multivariate Darboux method in enumeration problems, *J. of Combinatorial theory A*, 67 (1994), 169-184.

[DV] M. Delest, G. VIENNOT, Algebraic languages and polyominoes enumeration, In *Proc. X Colloquium on Automata Languages and Programming*, Lecture Notes in Computer Sciences, 173-181, 1983.

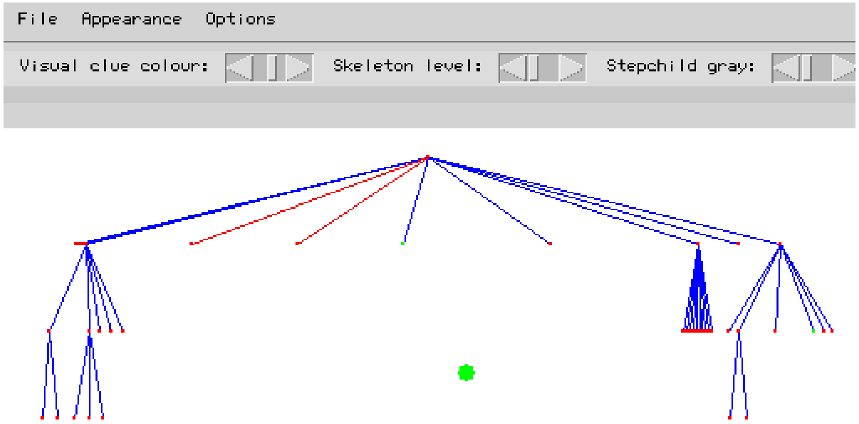
[De88] Generating functions for column-convex polyominoes, *J. of Comb. Th. A*, 4 **1** (1988), 12-31.

[De91] Polyominoes and animals : some recent results, *J. of Math. Chem.*, 8 (1991), 3-18.

[DF97] I. Dutour, J.M. F́dou, Object grammars and bijections, LaBRI publication 1164-97.



Tree before folding



Tree after folding confidence level 5%

Figure 11: Latour : a fold.

[Fl] P. Flajolet, Ambiguity and transcendence, In *Proc. 12th ICALP Colloquium*, 179-188, Lecture Notes in Computer Science, 194, 1985.

[FSZ] P. Flajolet, B. Salvy and P. Zimmermann, Automatic average-case analysis of algorithms, *Theoretical Computer Science*, 79 **1**, 1991.

[Gu] A.J. Guttmann, Planar Polygons: Regular, convex, almost convex, staircase and row convex, in *AIP Conference Proceedings* 248, 1992.

[HDM] I. Herman, M. Delest, G. Melançon, Tree visualization and navigation clues for information vizualization, *Computer Graphics Forum*, 17 **2** 1998, 153-165,.

[Ir61] E.T. Irons, A syntax directed compiler for Algol 60, *Comm. ACM*, 4 (1961), 51-55.

[Ir63] E.T. Irons, Towards more versatile mechanical translations, in *Proc. Symp. Appl. Math.*, 15 (1963), 41-50.

[Kn] D.E. Knuth, Semantics of context-free languages, *Maths. Sys.*, 2 (1968), 127-145.

[Ma] Waterloo Maple Company, http://www.maplesoft.com.

[Mu] Mupad distribution, http://www.mupad.de.

[Ni] M. Nivat, Transductions des langages de Chomsky, *Ann. Inst. Fourier*, 18 (1968), 339-456.

[Ri] J. Riordan, Combinatorial identies, Robert E. Krieger Publishing Company, 1979.

[S62] M.P. Schützenberger, Certain elementary families of automata, In *proc. Symp. on Mathematical Theory of Automata*, 139-153,Polytechnic Institute of Brooklyn, 1962.

[S63] M.P. Schützenberger, Contect-free languages and pushdown automata, Information and Control, **6** (1963), 246-264.

[S] N.J. Sloane, Sloane's On-Line Encyclopedia of Integer Sequences, http://www.research.att.com/ njas/sequences/eisonline.html.

[Vi91] X. Viennot, private communication with M.P. Schützenberger, Mai 1991.

[Vi90] X. Viennot, Trees, in *Mots, mlange offert M.P. Schützenberger*, 265- 291, Lothaire (ed.), Hermes, Paris, 1990.

[Vi92] X. Viennot, A survey of polyominoes enumeration, in *Actes SFCA'92*, 399-420, Ed. P. Leroux et C. Reutenauer, Montral, 1992.

[VV] M. Vauchaussade & X. Viennot, Enumeration of RNA secondary structures by complexity, in *Actes Mathematics in Biology and Medecine*, 360-365, Lecture Notes in Biomathematics, 1985.

[Wi] H.S. Wilf, Generatingfunctionology, Academic Press, 1994. luo gaia

[H] J. H. Hubbard, Local connectivity of Julia sets and bifurcation loci: three theorems of J. C. Yoccoz, In *Topological methods in modern mathematics*, 467-511, Publish or Perish Inc., 1993.

[MSS] R. Mañe, P. Sad & D. Sullivan, On the Dynamics of Rational Maps, *Ann. Scie. École Norm. Sup.* (4) **16** (1983), 193-217.