# Algorithms seminar, 1991-1992

Philippe Flajolet, Paul Zimmermann

UNITÉ DE RECHERCHE
INRIA-ROCQUENCOURT

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P. 105
78153 Le Chesnay Cedex
France
Tél.:(1)39 63 55 11

Rapports de Recherche

N°1779

***Programme 2***

*Calcul symbolique, Programmation*
*et Génie logiciel*

# ALGORITHMS SEMINAR, 1991-1992

**Philippe FLAJOLET**
**Paul ZIMMERMANN**

**Octobre 1992**

# ALGORITHMS SEMINAR, 1991–1992

Philippe Flajolet and Paul Zimmermann
(Editors)

**Abstract.** These seminar notes represent the proceedings, consisting of 36 brief articles (some in French), of a seminar devoted to the analysis of algorithms and related topics. The subjects covered include combinatorial models, generating functions and symbolic computation, asymptotic analysis, average case analysis of algorithms and data structures, and computational number theory.

# SÉMINAIRE ALGORITHMES, 1991-1992

**Résumé.** Ces notes de séminaire représentent le compte-rendu, constitué de 36 articles brefs, certains en anglais, d'un séminaire consacré à l'analyse d'algorithmes et aux sujets connexes. Les thèmes couverts comprennent: modèles combinatoires, séries génératrices et calcul formel, analyse asymptotique, analyse en moyenne d'algorithmes et de structures de données, ainsi que théorie algorithmique des nombres.

# ALGORITHMS SEMINAR, 1991–1992

PHILIPPE FLAJOLET AND PAUL ZIMMERMANN[1]
(Editors)

**Abstract**

These seminar notes represent the proceedings, consisting of 36 brief articles (some in French), of a seminar devoted to the analysis of algorithms and related topics. The subjects covered include combinatorial models, generating functions and symbolic computation, asymptotic analysis, average case analysis of algorithms and data structures, and computational number theory.

The primary goal of this seminar is to cover the major methods of the average case analysis of algorithms and data structures. In this capacity, the present book describes original analyses of digital trees, multidimensional search, quadtrees, suffix trees, heap ordered trees, index trees, merge sort, and string matching.

The underlying methods borrow from combinatorial enumerations, discrete probability, as well as asymptotic analysis, generating functions being central to most of these investigations. Several articles deal with combinatorial enumerations of classical combinatorial objects like binary partitions, Young tableaux, and polyominoes. Asymptotic methods include singularity analysis at large (Tauberian methods), the saddle point method, and functional equations like the ones related to divide–and–conquer recurrences.

Computer algebra is a neighbouring field that plays an increasingly important rôle in this area. It provides a collection of tools that permit to analyse complex models of combinatorics and the analysis of algorithms; at the same time, it inspires the quest for developing ever more systematic solutions to the analysis of well characterized classes of problems. In this vein, the notes contain several recent developments regarding the automatic computation of limits and the manipulation of recurrences or their generating function counterparts. The recent framework of holonomic functions forms the subject of several articles.

Finally, close to both algorithmic theory and computer algebra, the area of computational number theory has seen recently spectacular developments, some of it owing to its relevance to the design and analysis of modern cryptographic systems. We find here presentations relative to primality testing, sieve machines and Carmichael numbers next to the design of specialized circuitry for fast integer arithmetic.

The 36 articles included in this book represent snapshots of current research in these areas. A tentative organization of their contents is given below.

[1]The research reported here was supported in part by the ESPRIT II and ESPRIT III Basic Research Action Programmes of the E.C. under contracts #3075 and #7141 (projects ALCOM I and ALCOM II).

## PART I. COMBINATORIAL MODELS

In addition to its own traditions rooted in mathematics, the study of *combinatorial models* arises naturally in the process of analyzing algorithms that often involve classical combinatorial structures like strings, trees, graphs, permutations.
Young tableaux reflect several deep combinatorial properties of permutations, like the length of the longest increasing subsequence. Some of their generalizations lend themselves to elegant closed form formulæ [**1**]. Polyominoes [**2**] are closely related to lattice paths and, via encodings, they can be enumerated by formal languages, following a methodology originated by Schützenberger and well developed by Viennot and his school.
The analysis of maxima in convex regions of various shapes [**3**] illustrates the use of probabilistic methods in problems of combinatorial and stochastic geometry. Such methods, as we know from the examples of random graphs and combinatorial optimization problems, are very useful as soon as exact analytic models cease to be available.
The relations between algebraic methods and combinatorial models form the core of [**4**] and [**5**]. Random shuffles can be analyzed by viewing them as random walks on certain special groups [**4**]. In line with works of Allouche and Shallit, the algebra of simple divide–and–conquer recurrences now appears as a new facet of the older theory of regular languages and rational series [**5**].

[**1**] *Enumeration of Semi–Standard Young Tableaux,* D. Gouyou-Beauchamps

[**2**] *Counting Convex Polyominoes According to Their Area,* M. Bousquet-Mélou

[**3**] *Maxima in Convex Regions,* M. J. Golin

[**4**] *Fourier Transforms over Semi-simple Algebras,* F. Bergeron

[**5**] *Suites 2-régulières et séries rationnelles,* Ph. Dumas

## PART II. GENERATING FUNCTIONS AND SYMBOLIC COMPUTATION

Most exactly solvable models of combinatorics and analysis of algorithms rest on a suitable algebra of *generating functions.* Once this has been recognized, a reasonable aim is to find decision procedures for these classes of generating functions. Computer algebra systems provide a way of testing and implementing the methods, and the problem of optimizing the corresponding procedures often represents a non trivial problem of *symbolic computation.*
From extensive computations on close to 5000 integer sequences, forming the second edition of Sloane's *Handbook of Integer Sequences*, it appears that 23% are susceptible of a description found automatically by a dedicated computer algebra program [**6**]. The descriptions found can then usually be accounted for without too much pain by general theories of enumerations by generating functions. This "naturalist's study" provides unique statistical data regarding the power of general combinatorial frameworks based on generating functions.
The article [**7**] discusses the manipulation of algebraic objects (numbers and functions) in computer algebra systems. Algebraic functions are known to play an important rôle in combinatorial analysis as they occur in any problem that can be described by a context–free language.

An especially important class of generating functions is the class of functions that satisfy linear differential equations with polynomial coefficients, which, following Zeilberger, we call here *holonomic*. The corresponding sequences are also named holonomic, older terms being $D$–finite and $P$–recursive. In the analysis of algorithms, such sequences naturally occur in problems related to order statistics and search trees, a famous example being Quicksort itself. A formalization of the properties of holonomic functions and sequences has emerged from the works of Stanley, Lipshitz and Zeilberger.
The article [**8**] describes the basic closure properties following Stanley. The non trivial multivariate extension, due to independent approaches of Lipshitz and Zeilberger, is described in [**9**]. Gessel showed how to extend this framework to symmetric functions, from which there result non trivial recurrences and asymptotics for regular graphs, Young tableaux, Latin rectangles, etc, see [**10**].
Zeilberger showed how the class of holonomic functions permits to decide many combinatorial identities. Other properties of these functions, like their rational or algebraic character, are otherwise known to be decidable. The article [**11**] explains how differential algebra permits to determine closed form solutions for holonomic functions of low order.
The manipulation of functions, for our purposes often generating functions, is examined in [**12**]. In [**12**], a global approach based on algebraic differential equations is proposed for the manipulation of special functions in symbolic computation.
Finally, as already known from the experience gained with the Lambda–Upsilon–Omega system ($\Lambda\Upsilon^{\Omega}$), the manipulation of combinatorial generating functions can be automated for many classes of problems. The article [**13**] shows how a form of function composition can be embedded into an automatic analyzer programme, so that for instance iterated differentiation becomes automatically analyzable.


[**6**] *Approximations de séries génératrices,* S. Plouffe

[**7**] *Autour des nombres et fonctions algébriques en Maple,* M. Rybowicz

[**8**] *Introduction aux fonctions holonomes en une variable,* Ph. Flajolet

[**9**] *Fonctions holonomes à plusieurs variables,* K. Compton

[**10**] *Holonomic Symmetric Functions,* D. Gouyou-Beauchamps

[**11**] *L'algorithme de Kovačić,* M. Loday-Richaud

[**12**] *Functions in Symbolic Computation,* J. R. Shackell

[**13**] *Function Composition and Automatic Average-Case Analysis,* P. Zimmermann


## PART III. ASYMPTOTIC ANALYSIS

*Asymptotic analysis* is an essential ingredient in the interpretation of quantitative results supplied by the resolution of combinatorial models. An important class of problems involves recovering the asymptotic form of the coefficients of a function from asymptotic properties of the function itself. Tauberian theory may be used to relate the dominant positive singularity of a generating function to the form of its Taylor coefficients. The article [**14**] shows how Wiener's general Tauberian theorem permits to extract coefficients asymptotically; this is useful even in the possible presence

of natural boundaries where methods based on contour integrals may fail. This approach represents a useful complement to singularity analysis or Darboux's method in such cases. The saddle point method tends to apply well to entire functions; in [**15**] we find a presentation of its use in analysing coefficients of large powers of generating functions that occur in enumerations of various sets, sequences or forests, as well as some forms of the central limit theorem.
Divide-and-conquer recurrences, like the mergesort recurrence, require an incursion into the realm of Dirichlet generating functions and Mellin transforms; in this way top down recursive mergesort gets fully analyzed [**16**]. A combination of saddle point and Mellin transform techniques is needed to estimate enumeration counts relative to binary partitions and certain sequences of the divide-and-conquer type that exhibit superpolynomial growth [**17**].
The special problem of estimating the fractional part of $(\frac{3}{2})^n$, see [**18**], lies at the crossroads of special functions, asymptotic analysis, and number theory (Waring's problem). The localization of zeroes of polynomials is a delicate task that crucially intervenes in the analysis of coefficients of rational functions [**19**].
Finally, larger and larger classes of asymptotic problems can now be treated automatically using computer algebra. The manipulation of asymptotic scales of a very general nature in Hardy fields forms the subject of [**20**]. Such studies will be eventually useful in dealing with saddle point Cauchy integrals or generalized versions of the singularity analysis theory.


[**14**] *Théorèmes taubériens pour l'énumération asymptotique,* K. Compton

[**15**] *The Asymptotic Behaviour of Coefficients of Large Powers of Functions,* D. Gardy

[**16**] *Transformée de Mellin et asymptotique : le tri-fusion,* M. Golin

[**17**] *Asymptotique de récurrences et dénombrement de partitions,* Ph. Dumas

[**18**] *Minorations de* $\|(3/2)^k\|$, L. Habsieger

[**19**] *La recherche des racines complexes d'un polynôme selon Schönhage,* X. Gourdon

[**20**] *Algorithms for Computing Limits and Asymptotic Forms,* J. R. Shackell


## PART IV. ANALYSIS OF ALGORITHMS AND DATA STRUCTURES

This section deals with the *analysis of algorithms* and *data structures*. Trees have been recognized by various authors as the single most important structure in computer science. Not unnaturally, several analyses found here are devoted to tree structures.
Article [**21**] introduces a very general model of increasing trees that includes quicksort, binary search trees, and heap ordered trees. The analysis relies on singularity analysis of a non linear ordinary differential equation. Differential equations are also known to rule quadtrees, and a singularity perturbation analysis is employed in order to prove a Gaussian limit for the distribution of search costs in such trees [**22**].
Digital structures, tries or digital search trees, lead to difference equations [**23**] that serve to analyze corresponding multidimensional search problems [**24**]. Such trees actually offer attractive highly practical perspectives when implementing a compact index [**25**].
The last three articles in this section deal with pattern matching in strings. The average case analysis of string matching is discussed in [**26**]. Some questions in this range are closely related to

digital tries, via the suffix tree structure [**27**]. Finally, some of the probabilistic intuition gleaned when analyzing one dimensional problems can be used to speed up pattern matching in planar data [**28**].

[**21**] *Variétés d'arbres croissants,* B. Salvy

[**22**] *Limit Distributions in Quadtrees,* Th. Lafforgue

[**23**] *Arbres digitaux et équations aux différences,* Ph. Flajolet

[**24**] *Multidimensional Digital Searching,* H. Prodinger

[**25**] *Compact Balanced Tries,* P. Nicodème

[**26**] *Performances d'algorithmes de recherche de motifs,* M. Régnier

[**27**] *Analyse des arbres suffixes par motif coulissant,* Ph. Jacquet

[**28**] *Fast Two Dimensional Pattern Matching,* M. Régnier

## PART V. COMPUTATIONAL NUMBER THEORY

Primality testing and factorization are amongst the oldest problems of *computational number theory.* Diophantine equations come next. Sieve machines can be used to solve a variety of such arithmetical problems, and a historical perspective is given in [**29**]. Primality testing has made noticeable progress since the years 1970, the fastest algorithms trading rigour for speed and being therefore probabilistic [**30**]. An interesting collection of hard cases for primality tests of Fermat type are the Carmichael numbers [**31**]. It was proved only in 1992 that there are infinitely many Carmichael numbers.
The next two articles consider the design of fast circuits dedicated to basic operations on numbers. One centres on a unified method to perform divisions, square roots, and other elementary functions in hardware [**32**]. The other one [**33**] reveals ideas based on 2–adic numbers that enter the design of a chip implementing the RSA scheme.
A cryptanalysis of the DES forms the subject of [**34**].
Finally, the last two brief summaries, [**35**] and [**36**] point to interesting relations between logic, number theory and computing.

[**29**] *Histoire et application des machines de crible numérique,* H. C. Williams

[**30**] *Probabilistic Primality Testing,* A. O. L. Atkin

[**31**] *Nombres de Carmichael,* D. Guillaume

[**32**] *Algorithmes pour la conception de circuits arithmétiques rapides,* J-M. Muller

[**33**] *Circuits synchrones, nombres 2-adiques, et codages RSA,* J. Vuillemin

[**34**] *Cryptanalyse différentielle du DES en* 16 *rounds,* A. Shamir

[**35**] *Primitive Recursive Functions and Exponential Diophantine Equations,* Y. Matijasevich

[**36**] *Some Investigations on the Riemann Hypothesis with Computers,* Y. Matijasevich

**Acknowledgements.** The lectures summarized here emanate from a community of researchers in the analysis of algorithms, in the ALGORITHMS Project at INRIA and in the greater Paris area—especially École Polytechnique (J.–M. Steyaert), University of Paris Sud at Orsay (D. Gouyou Beauchamps, D. Gardy) and LITP (M. Soria).
The editors express their gratitude to the various persons who supported actively this joint enterprise, most notably: Abalo Baya, Kevin Compton, Philippe Dumas, Danièle Gardy, Mordecai Golin, Dominique Gouyou–Beauchamps, François Morain, Pierre Nicodème, Bruno Salvy, Michèle Soria, Jean–Marc Steyaert. Many thanks are due also to the speakers and to the authors of summaries.
We are especially indebted to François Morain and Bruno Salvy for their devoted help in organizing this seminar, to Michèle Soria for much appreciated assistance in the editorial job, and to Virginie Collette for permanently keeping the wheel in motion.

The Editors
PH. FLAJOLET AND P. ZIMMERMANN

# Part I

# Combinatorial Models

# 1

# Enumeration of Semi–Standard Young Tableaux

Dominique Gouyou-Beauchamps
LRI, Orsay

[summary by Philippe Flajolet]

**Standard tableaux.** A *Young tableau* (also known as a *standard tableau*) is an arrangement of distinct integers in an array of left justified rows, such that the entries of each row are in increasing order from left to right and the entries of each column are in increasing order from bottom to top[2]. For instance, here is an array of size 15:

$$\begin{array}{cccccc} 11 & & & & & \\ 4 & 8 & 12 & 14 & & \\ 3 & 6 & 7 & 13 & & \\ 1 & 2 & 5 & 9 & 10 & 15\,. \end{array}$$

In general a tableau of $n$ cells has a shape described by a partition of the integer $n$, the shape of the example being $(6,4,4,1)$.
Such tableaux originate in the study of linear representations of the symmetric group. The main result is the following: *Each permutation of* $[1..n]$ *is uniquely representable by a pair of tableaux of size* $n$ *having the same shape.* The effective mapping—based on the "bumping rule"—constitutes the celebrated Robinson–Schensted correspondence. A very readable introduction to the subject is to be found in Knuth's book [4, Sec. 5.1.4].
Many properties result from the Robinson–Schensted correspondence. First, the total number of tableaux of size $n$ is equal to the number of involutions (these are permutations $\tau$ such that $\tau^2 = Id$) of $[1..n]$, that is

$$T_n = n! \cdot [z^n] \exp(z + \frac{z^2}{2}).$$

Next, the length of the longest increasing subsequence of a permutation $\tau$ coincides with the common base size of the pair of tableaux associated with $\tau$.
The enumeration of tableaux of bounded width or height thus appears of interest in connection with various order statistics, these structures being also rich combinatorial objects *per se*. Let $T_n^{[k]}$ be the number of tableaux of height $\leq k$, and denote by $C_n = 1/(n+1)\binom{2n}{n}$ the Catalan number. Then, from the works of Regev and Gouyou–Beauchamps, we have

$$T_n^{[2]} = \binom{n}{\lfloor n/2 \rfloor}; \quad T_n^{[3]} = \sum_{i=0}^{\lfloor n/2 \rfloor} \binom{n}{2i} C_i; \quad T_n^{[4]} = C_{\lfloor (n+1)/2 \rfloor} C_{\lceil (n+1)/2 \rceil}. \tag{1}$$

The formulæ become intricate as $k$ increases. However, it is known that the quantities $T_n^{[k]}$ are holonomic for each fixed $k$. See Gessel's paper on this aspect of things [3].

[2] There is a concurrent tradition of representing such arrays downwards!

**Semi–standard tableaux.** This part deals with results obtained jointly by Gouyou–Beauchamps and Seul Hee Choi in a paper yet to appear [2]. They are relative to *semi–standard* tableaux. A semi–standard tableau is strictly increasing in rows but only weakly increasing in columns. Here is one:

$$\begin{array}{llllll} 4 & & & & & \\ 2 & 3 & 5 & & & \\ 1 & 2 & 4 & & & \\ 1 & 2 & 3 & 4 & 5\,. \end{array} \tag{2}$$

B. Gordon had proved long ago (in the 1960's) that the number of such tableaux with at most $r$ rows (height at most $r$) and entries between 1 and $n$ admits a product form

$$a_{n,r} = \prod_{1\le i\le j\le n} \frac{r+i+j-1}{i+j-1}. \tag{3}$$

This looks much simpler than what (1) reveals of the corresponding problem for standard tableaux. Choi and Gouyou–Beauchamps obtain a new formula that refines on (3) by keeping track of the parity of columns.

**Theorem 1** *The number of semi–standard Young tableaux with entries in* $[1..n]$ *having height at most* $2k$ *and having* $p$ *columns of odd height is*

$$c_{n,2k,p} = \frac{\binom{n}{p}\binom{2k+p-1}{p}}{\binom{n+2k+p}{p}} \prod_{1\le i\le j\le n} \frac{2k+i+j}{i+j}.$$

The numbers $c_{n,2k,0}$ had been earlier determined by Desainte–Catherine and Viennot, and the work presented here constitutes a refinement of their method. The ingredients of the proof are as follows.
1. Semi–standard tableaux are bijectively equivalent to certain non negative matrices and to certain generalized involutions (Knuth 1970, Burge 1974).
2. Semi–standard tableaux are bijectively equivalent to special pilings of Dyck paths. (A Dyck path is a non negative gambler ruin sequence.)
3. The counting of the relevant Dyck path configurations is expressible in terms of determinants involving ballot numbers and Hankel determinants of Catalan numbers. For instance,

$$\begin{vmatrix} C_0 & C_1 & \cdots & C_{n-1} & C_n \\ C_1 & C_2 & \cdots & C_n & C_{n+1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ C_{n-1} & C_n & \cdots & C_{2n-2} & C_{2n-1} \\ C_n & C_{n+1} & \cdots & C_{2n-1} & C_{2n} \end{vmatrix} = 1.$$

(This part involves the Gessel–Viennot theory of path determinants.)
4. The involved determinants can finally be calculated using Viennot's combinatorial theory of the $qd$ algorithm.
The whole enterprise is a rather delicate (and intricate) piece of bijective combinatorics. The effort invested is also justified by a general attempt at understanding the combinatorics of Gordon's amazingly simple formula (1). Initially, it arose as a specialization of a $q$–analog¡, itself related to plane partitions—that is, generalized tableaux classified according to the sum of their elements. Andrews' book can be consulted on some of these aspects, see [1, Ch. 11]. Plane partitions are lurking in the background.

## References

[1] G. E. Andrews. *The Theory of Partitions*, volume 2 of *Encyclopedia of Mathematics and its Applications*. Addison–Wesley, 1976.

[2] S. H. Choi and D. Gouyou-Beauchamps. Énumeration de tableaux de Young semi-standards. To appear in *Theoretical Computer Science*, 1992.

[3] I. M. Gessel. Symmetric functions and $P$–recursiveness. *Journal of Combinatorial Theory,* Series A, 53:257–285, 1990.

[4] D. E. Knuth. *The Art of Computer Programming*, volume 3 : Sorting and Searching. Addison-Wesley, 1973.

# 2

# Counting Convex Polyominoes According to Their Area

Mireille Bousquet-Mélou
LaBRI, Université de Bordeaux I

[summary by Dominique Gouyou-Beauchamps]

Two approaches are presented for the enumeration according area of different classes of convex polyominoes. The first approach is based on the concept of coins stacking. In the second approach, the elementary decomposition of polyominoes leads, for each class, to a $q$-equation, that is sometimes solvable.

Unit squares with vertices at integer points in the cartesian plane are called *cells*. A *polyomino* is a finite connected union of cells such that the interior is also connected (no cut point). The *area* of a polyomino is the number of cells, the *perimeter* is the length of the border. Polyominoes are defined up to a translation. A *column* (resp. *row*) of a polyomino is the intersection between the polyomino and any infinite vertical (resp. horizontal) strip of unit squares. A polyomino is said to be *column-* (resp. *row-*) *convex* if all its columns (resp. rows) are connected. A *convex* polyomino is both row- and column-convex.

Let $P$ a convex polyomino and $Rect(P)$ be the smallest *rectangle* (considered as a convex polyomino) containing $P$. The polyomino touches the border of $Rect(P)$ along four connected segments. Each of these segments has two extreme points and thus we introduce 8 points, as shown in Figure 1. The Westmost (respectively Eastmost) of the points of $P$ containing the South (respectively North) border of $Rect(P)$ is denoted by $S(P)$ (respectively $N(P)$). Following counterclockwise the border of $P$, one meets successively the above 8 canonical points in the order: $S(P)$, $S'(P)$, $E(P)$, $E'(P)$, $N(P)$, $N'(P)$, $W(P)$, $W'(P)$. The *height* and the *length* of the convex polyomino $P$ are the height and the length of the rectangle $Rect(P)$ (see Figure 1). We can now define several important subclasses of convex polyominoes. A *parallelogram polyomino* is a convex polyomino $P$ such that $S(P) = W'(P)$ and $N(P) = E'(P)$ (see Figure 2). A *stack polyomino* is a convex polyomino such that $S(P) = W'(P)$ and $S'(P) = E(P)$ (see Figure 2). A *Ferrers diagram* is a convex polyomino $P$ such that $N(P) = E'(P)$, $S'(P) = E(P)$ and $S(P) = W'(P)$ (see Figure 2). A *directed convex polyomino* is a convex polyomino $P$ such that $N(P) = E'(P)$ (see Figure 2). Polyominoes are very classical objects in combinatorics. Counting polyominoes according to their area or perimeter is a major unsolved problem in combinatorics. The first authors interested in this subject were Read [8] and Golomb [7]. Physicists have given several asymptotic results. They call *animal* a set of points obtained by taking the centers of the cells of a polyomino. Giving a privileged direction for the growth of an animal allows them to obtain generating functions (see Viennot [9] and references therein). In theoretical computer science, Yuba and Hoshi [11] introduced directed polyominoes (or animals) for a new method for key searching. They consider that a polyomino is a *binary search network* structure.

The following sections give some enumeration results for different classes of polyominoes.

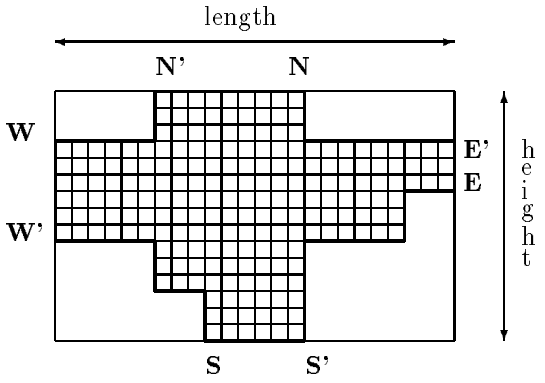



Figure 1: A convex polyomino

# 1 Counting convex polyominoes according to their length and their height

**Theorem 1** *The generating function for the number $p_{m,n}$ of convex polyominoes having height $m$ and length $n$ is:*

- $\sum_{m,n} p_{m,n} x^m y^n = \frac{xy}{1-x-y}$ *for Ferrers diagrams,*
- $\sum_{m,n} p_{m,n} x^m y^n = \frac{xy(1-x)}{1-2x-y+x^2}$ *for stack polyominoes,*
- $\sum_{m,n} p_{m,n} x^m y^n = \frac{1-x-y-\sqrt{\Delta}}{2}$ *for parallelogram polyominoes,*
- $\sum_{m,n} p_{m,n} x^m y^n = \frac{xy}{\sqrt{\Delta}}$ *for directed convex polyominoes (Chang and Lin [3]),*
- $\sum_{m,n} p_{m,n} x^m y^n = \frac{xy}{\Delta^2}(1-3x-3y+3x^2+3y^2+5xy-x^3-y^3-x^2y-xy^2-xy(x-y)^2) - \frac{4x^2y^2}{\Delta^{3/2}}$ *for convex polyominoes (Chang and Lin [3], Delest and Viennot [4]), where* $\Delta = 1-2x-2y-2xy+x^2+y^2$.

# 2 Counting convex polyominoes according to their length, their height and their area

**Theorem 2** *The generating function for the numbers $f_{m,n,a}, s_{m,n,a}$ of convex polyominoes having height $m$, length $n$ and area $a$ is:*

- $\sum_{m,n,a} f_{m,n,a} x^m y^n q^a = \sum_{m \geq 1} \frac{xy^m q^m}{(yq)_m}$ *for Ferrers diagrams,*
- $\sum_{m,n,a} s_{m,n,a} x^m y^n q^a = \sum_{m \geq 1} \frac{xy^m q^m}{(yq)_{m-1}(yq)_m}$ *for stack polyominoes, where* $(a)_n = (1-a)(1-aq)(1-aq^2)...(1-aq^n)$ *and* $(a)_0 = 1$.

In [5] and [6], Delest and Fédou give the generating function for the number of parallelogram polyominoes according to their length and their area.

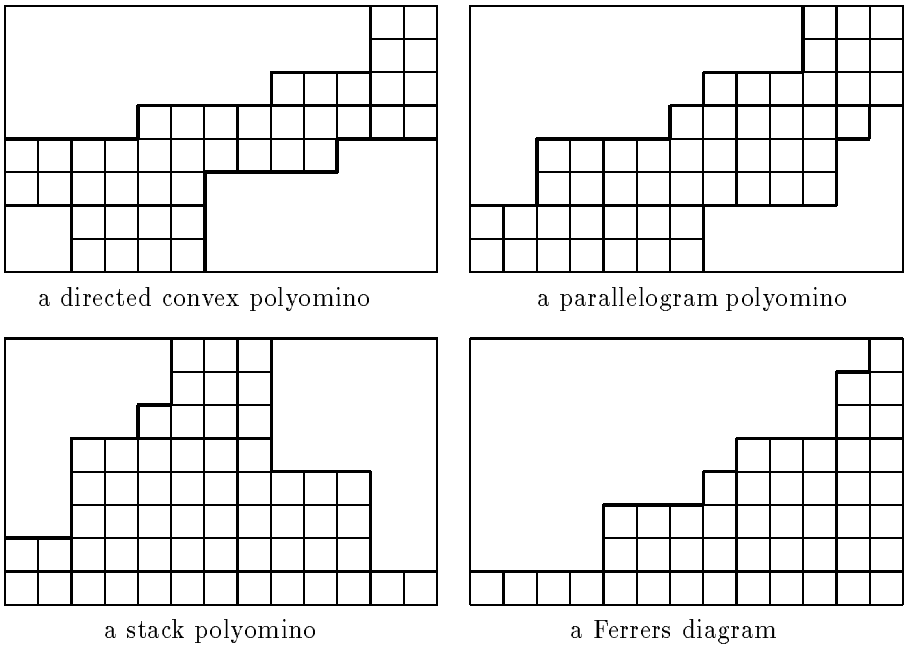



Figure 2: Different classes of convex polyominoes

## 3 Stack polyominoes

**Theorem 3** *The generating function for the number $s_{m,n,a}$ of stack polyominoes having height $m$, length $n$ and area $a$ is:*

$$\sum_{m,n,a} s_{m,n,a} x^m y^n q^a = \sum_{m\geq 1} \frac{xy^m q^m}{(yq)_{m-1}(yq)_m} = \sum_{n\geq 1} \frac{xy^n q^n T_n}{(xq)_n},$$

*where*

$$(a)_n = (1-a)(1-aq)(1-aq^2)...(1-aq^n) \text{ and } (a)_0 = 1,$$

$$T_n = 1 + \sum_{2\leq 2k\leq n} x^k q^{k^2} \sum_{m=k}^{n-k} \left[ \begin{array}{c} m \\ k \end{array} \right]_q \left[ \begin{array}{c} n-m-1 \\ k-1 \end{array} \right]_q,$$

$$\left[ \begin{array}{c} n \\ k \end{array} \right]_q = \frac{[n]!}{[k]![n-k]!} \text{ and } [n]! = 1(1+q)(1+q+q^2)...(1+q+...+q^{n-1}).$$

## 4 Parallelogram polyominoes

**Theorem 4** *The generating function for the number $p_{m,n,a}$ of parallelogram polyominoes having height $m$, length $n$ and area $a$ is:*

$$\sum_{m,n,a} p_{m,n,a} x^m y^n q^a = y\frac{N_1}{N} \text{ where } N = \sum_{n\geq 0} \frac{(-1)^n x^n q^{\binom{n+1}{2}}}{(q)_n (yq)_n} \text{ and } N_1 = \sum_{n\geq 1} \frac{(-1)^{n+1} x^n q^{\binom{n+1}{2}}}{(q)_{n-1}(yq)_n},$$

$$(a)_n = (1-a)(1-aq)(1-aq^2)...(1-aq^n),\ n \geq 1 \text{ and } (a)_0 = 1.$$

The proof uses the concept of *heaps of pieces* introduced by Viennot [10] (see also [2]).

**Theorem 5** *The generating function for the number $p_{h,m,n,a}$ of parallelogram polyominoes having height $m$, length $n$, area $a$ and their first (or last) column of height $h$ is:*

$\sum_{m,n,a} p_{h,m,n,a} x^m y^n q^a = x y^h q^h \frac{N(xq^h)}{N(x)}$ *where, as in Theorem 4,* $N(x) = \sum_{n\geq 0} \frac{(-1)^n x^n q^{\binom{n+1}{2}}}{(q)_n (yq)_n}$.

## 5 Directed convex polyominoes

**Theorem 6** *The generating function for the number $d_{m,n,a}$ of directed convex polyominoes having height $m$, length $n$ and area $a$ is:*

$$\sum_{m,n,a} d_{m,n,a} x^m y^n q^a = y\frac{N_1+N_2}{N} \text{ where } N = \sum_{n\geq 0} \frac{(-1)^n x^n q^{\binom{n+1}{2}}}{(q)_n (yq)_n},\ N_1 = \sum_{n\geq 1} \frac{(-1)^{n+1} x^n q^{\binom{n+1}{2}}}{(q)_{n-1} (yq)_n}$$

$$\text{and } N_2 = y \sum_{n\geq 2} \left( \frac{x^n q^n}{(yq)_n} \sum_{m=0}^{n-2} \frac{(-1)^m x^n q^{\binom{m+2}{2}}}{(q)_m (yq^{m+1})_{n-m-1}} \right).$$

The fundamental idea of the proof is to split a directed convex polyomino into two simpler polyominoes (see Figure 3).

## 6 Convex polyominoes

**Theorem 7** *The generating function for the number $c_{m,n,a}$ of convex polyominoes having height $m$, length $n$ and area $a$ is:*

$$Z(x,y,q) = \sum_{m,n,a} c_{m,n,a} x^m y^n q^a = 2y\frac{(N_1+N_2)N_3}{N} - 2yZ_1 - Z_2$$

*where $N$, $N_1$ and $N_2$ are defined in Theorem 6 and where*

$$N_3 = \sum_{m\geq 0} \frac{xq^{m+1} T_m M_{m+1}}{(xq)_m},\ Z_1 = \sum_{0\leq n\leq m} \frac{M_{m+1}^{n+1} T_m T_n}{(xq)_{n-1} (xq)_n},\ Z_2 = \sum_{n\geq 1} \frac{xy^n q^n (T_n)^2}{(xq)_{n-1} (xq)_n}$$

$$T_0 = T_1 = 1,\ T_n = 2T_{n-1} + (xq^{n-1} - 1) T_{n-2},$$

$$M_0 = 0,\ M_1 = 1,\ M_n = (1 + y - xq^{n-1}) M_{n-1} - yM_{n-2},$$

$$M_n^n = -xy^{n-1} q^n,\ M_m^n = x^2 y^{n-1} q^{n+m} M_{m-n}(xq^n) \text{ if } m > n.$$

The fundamental idea of the proof is to split a convex polyomino into three simpler polyominoes as in [4]. With a bisection we obtain the following expression for $Z(x,y,q)$:

$$Z(x,y,q) = 2y \sum_{m\geq 1} \frac{y^{m+2}(T_{m+1} S(xq^m) - yT_m S(xq^{m+1}))^2}{[(xq)_m]^2 N(xq^{m-1}) N(xq^m)} + \sum_{m\geq 1} \frac{xy^m q^m (T_m)^2}{(xq)_{m-1} (xq)_m},$$

$$\text{where } S(x) = \sum_{n\geq 1} \left( \frac{x^n q^n}{(yq)_n} \sum_{j=0}^{n-1} \frac{(-1)^j q^{\binom{j}{2}}}{(q)_j (yq^{j+1})_{n-j}} \right).$$

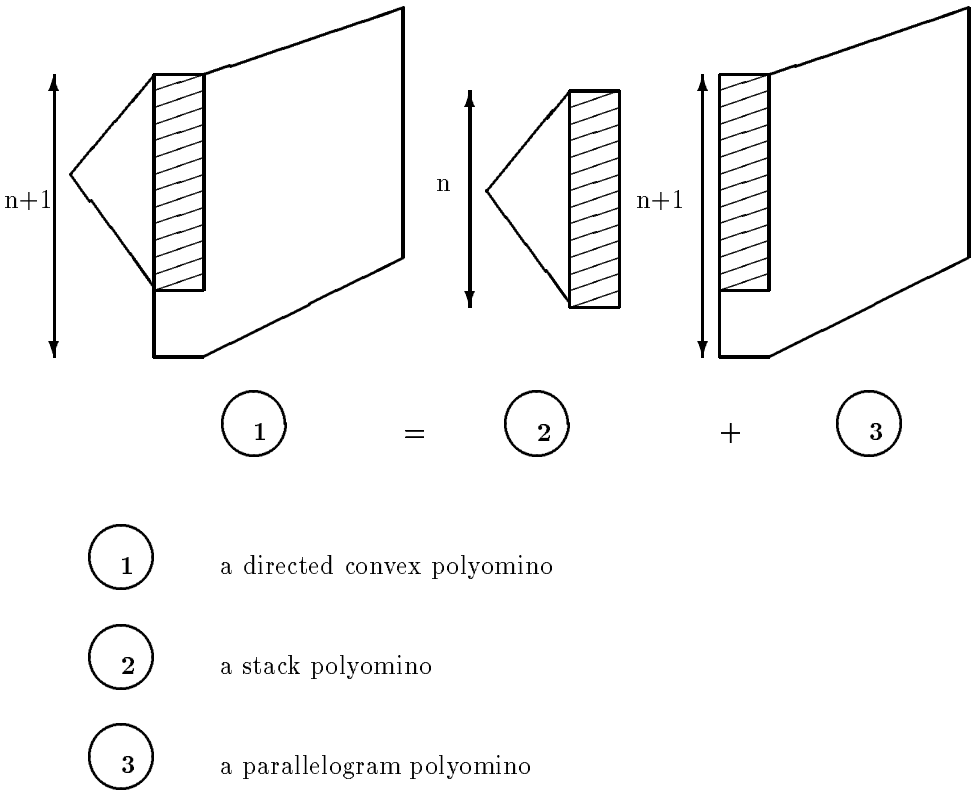



Figure 3: Decomposition of a directed convex polyomino

## 7 Functional equations

**Theorem 8** *The generating function* $P(t) = P(s,t,x,y,q) = \sum_{m,n,a} p_{u,v,m,n,a} s^u t^v x^m y^n q^a$ *of parallelogram polyominoes, having height* $m$*, length* $n$*, area* $a$*, their last column of height* $u$ *and their first column of height* $v$*, satisfies the following equation:*

$$P(t) = x\frac{styq}{1-styq} + x\frac{tq}{(1-tq)(1-tyq)}(P(1) - P(tq)).$$

This equation can be easily solved. The solution for $s = y = 1$ gives a result of Delest and Fédou [5] (enumeration according to area and number of columns) and for $s = 1$, $x = y = z^2$, gives a result of Guttman (enumeration according to area and perimeter).

**Theorem 9** *The generating function* $Y(t) = Y(s,t,x,y,q) = \sum_{m,n,a} y_{u,v,m,n,a} s^u t^v x^m y^n q^a$ *of directed convex polyominoes, having height* $m$*, length* $n$*, area* $a$*, their last column of height* $u$ *and their first column of height* $v$*, satisfies the following equations:*

$$Y(t) = \frac{xstyq}{1-styq} + tyqT(t) + \frac{xtq}{(1-tq)(1-tyq)}(Y(1) - Y(tq)) \text{ where } T(t) = \sum_{m\geq 2} \frac{x^m q^m yts}{(tyq)_{m-1}(1-tsyq^m)},$$

$$\text{and also } Y(s) = xstyq\frac{1-syq}{1-styq} + syqP(s) + \frac{xsq}{1-sq}Y'(1) + \frac{xs^2q^2}{(1-sq)^2}(Y(sq) - Y(1)).$$

**Theorem 10** *The generating function* $Z(s) = Z(s,t,x,y,q) = \sum_{m,n,a} z_{u,v,m,n,a} s^u t^v x^m y^n q^a$ *of convex polyominoes, having height* $m$*, length* $n$*, area* $a$*, their last column of height* $u$ *and their first column of height* $v$*, satisfies the following equation:*

$$Z(s) = xstyq\frac{1-2syq}{1-styq} + s^2y^2q^2T(t,s) + 2syqY(t,s) + \frac{xsq}{1-sq}Z'(1) + \frac{xs^2q^2}{(1-sq)^2}(Z(sq) - Z(1)).$$

# References


[1] M. Bousquet-Mélou. *q-Énumération de polyominos convexes.* Thèse de doctorat, Université de Bordeaux, 1991. Also available as LaCIM publication number 9, Montréal, 1991.

[2] P. Cartier and D. Foata. *Problèmes combinatoires de commutation et réarrangements*, volume 85 of *Lecture Notes in Mathematics.* Springer Verlag, 1969.

[3] S. J. Chang and K. Y. Lin. Rigorous results for the number of convex polygons on the square and honeycomb lattices. *Journal of Physics* Series A, 21:2635–2642, 1988.

[4] M. Delest and X. Viennot. Algebraic languages and polyominoes enumeration. *Theoretical Computer Science*, 34:169–206, 1984.

[5] M. P. Delest and J. M. Fédou. Enumeration of skew Ferrers diagrams. To appear in Discr. Math.

[6] M. P. Delest and J. M. Fédou. Attribute grammars are useful for combinatorics. Technical Report 122, University of Bordeaux I, 1989.

[7] S. Golomb. *Polyominoes.* Scribner's, New York, 1965.

[8] R. C. Read. Contributions to the cell growth problem. *Canadian Journal of Mathematics*, 14:1–20, 1962.

[9] X. G. Viennot. Problèmes combinatoires posés par la physique statistique. *Astérisque*, 121-122:225–246, 1985. Séminaire Bourbaki, $36^e$ année, $n^o$ 626, 1983-1984.

[10] X. G. Viennot. Heaps of pieces I: Basic definitions and combinatorial lemmas. In G. Labelle and P. Leroux, editors, *Combinatoire énumérative*, number 1234 in Lecture Notes in Mathematics. Springer-Verlag, 1986.

[11] T. Yuba and M. Hoshi. Binary search networks: a new method for key searching. *Information Processing Letters*, 24:59–65, 1987.

# 3

# Maxima in Convex Regions

Mordecai J. Golin
INRIA, Rocquencourt

[summary by Pierre Nicodème]

Suppose that $C$ is a bounded convex planar region. Let $p_1, \ldots, p_n$ be $n$ points drawn independently identically distributed from the uniform distribution over $C$ and let $M_n^C$ be the number of the points which are maximal. We present results showing how the asymptotic behaviour of $\mathbf{E}\left(M_n^C\right)$ depends on the geometry of $C$.

## 1 Introduction

We discuss $\mathbf{E}\left(M_n^C\right)$, the expected number of maximal points (that is such that there is no other point in the North-East quadrant from this point) in a set of $n$ Independently Identically Distributed (I.I.D.) points drawn from the uniform distribution over some convex bounded planar region $C$.
The corresponding question for convex hulls has been well studied. Renyi and Sulanke [7, 8] proved that if $n$ points are chosen I.I.D. from $C$ then, if $C$ is a convex polygon, the expected number of convex hull points is $\Theta(\log n)$ while if $C$ is convex and has a doubly continuously differentiable boundary the answer is $\Theta\left(n^{1/3}\right)$. Dwyer [4] provides a survey of more recent results.
The expected number of maxima has not been examined nearly as closely. It has been known for many years [1] that if $C$ is the unit square then $\mathbf{E}\left(M_n^C\right) = \Theta(\log n)$. Recently Dwyer [4] proved that $\mathbf{E}\left(M_n^C\right) = \Theta(\sqrt{n})$ when $C$ is a circle and also proved a general upper bound $\mathbf{E}\left(M_n^C\right) = O(\sqrt{n})$ that is valid for any bounded convex planar region $C$ (this result can also be found in Devroye [3]).
In this paper we study the asymptotics of $\mathbf{E}\left(M_n^C\right)$ in detail. Note that the condition that $C$ be convex is important. If it is abandoned then it can be shown that, for all functions $f$, $f(n) \leq n/\log^2 n$, and $f$ slowly varying at infinity, there is some $C$ such that $\mathbf{E}\left(M_n^C\right) = \Theta(f(n))$ (see [6] for details). If $C$ is constrained to be convex the situation is very different. We show in this paper that for convex $C$ either $\mathbf{E}\left(M_n^C\right) = \Theta\left(\sqrt{n}\right)$ or $\mathbf{E}\left(M_n^C\right) = O\left(\log n\right)$; nothing between these two functions is possible. We also give sufficient conditions for $\mathbf{E}\left(M_n^C\right) = \Theta(1)$ and $\mathbf{E}\left(M_n^C\right) = \Theta(\log n)$.
The rest of the paper will use the following notation: if $p = (p.x, p.y)$ and $q = (q.x, q.y)$ are planar points we say that $p$ *dominates* $q$ if $p.x \geq q.x$ and $p.y \geq q.y$. If $S = \{p_1, \ldots, p_n\}$ is a set of points we say that $p$ is *maximal* in $S$ if there is no $q \in S$, $q \neq p$, such that $q$ dominates $p$. We set

$$\mathrm{MAX}(S) = \{p : p \text{ is maximal in } S\}.$$

See Figures 1 (a), (b) and (c). Suppose $C$ is a bounded planar region. Let $S = \{p_1, \ldots, p_n\}$ be a set of $n$ points drawn I.I.D. from the uniform distribution over $C$. Let $M_n^C = |\text{MAX}(S)|$ be the number of maximal points in $S$. We will study $\mathbf{E}\left(M_n^C\right)$, the expected number of maximal points. We will also look at the *outer layer* of $S$, the set of all $p \in S$ such that one of the four quadrants of the Cartesian axes centered at $p$ contains no point in $S$ (see [3] for more details). A set's outer layer always contains the set's convex hull. We define $OL(S)$ to be the outer layer points of $S$ and $OL_n^C = |OL(S)|$ to be the number of such points.

## 2 Results

In what follows we will always assume that $C$ is a *closed* region. We do this to ensure that $C$ contains its boundary, $\partial C$ : this assumption makes our proofs slightly simpler. Notice though that the assumption is not restrictive. If $C$ is *any* bounded convex region then $\mathbf{E}\left(M_n^C\right) = E\left(M_n^{\overline{C}}\right)$ because a point chosen from the uniform distribution over $\overline{C}$ is in $\partial C$ with probability zero. It thus suffices to analyze $\mathbf{E}\left(M_n^C\right)$ for closed $C$. Our first theorem is

**Theorem 1 (The Gap Theorem)** *Let $C$ be a planar convex region. We say that a point $p \in C$ is an* upper-right-hand-corner *of $C$ if $p$ dominates every point $q \in C$. The expected number of maxima among $n$ points chosen I.I.D. uniformly from $C$ is qualitatively dependent upon whether $C$ has an upper-right-hand-corner:*

- *If $C$ does not have such a corner then $\mathbf{E}\left(M_n^C\right) = \Theta(\sqrt{n})$.*
- *If $C$ does have such a corner then $\mathbf{E}\left(M_n^C\right) = O(\log n)$.*

Note that for all convex $C$, $\mathbf{E}\left(M_n^C\right)$ can not behave like a function asymptotically between $\log n$ and $\sqrt{n}$. Hence the name Gap Theorem, alluding to the gap between the two possible behaviours.

**Example 1** Figures 1 (b), (c), (d), (f), and (h) all have upper-right-hand-corners and thus have $\mathbf{E}\left(M_n^C\right) = O(\log n)$ : figures 1 (a), (e), and (g) don't and so have $\mathbf{E}\left(M_n^C\right) = \Theta(\sqrt{n})$.

Recall that $OL_n^C$ is the number of outer-layer points among $n$ points chosen I.I.D. uniformly from $C$. Theorem 1 can be used to say quite a lot about the expected number of outer layer points.

**Corollary 1** *Let $C$ be a planar convex region. If $C$ is a rectangle with sides parallel to the Cartesian axes then $\mathbf{E}\left(OL_n^C\right) = \Theta(\log n)$. Otherwise $\mathbf{E}\left(OL_n^C\right) = \Theta(\sqrt{n})$.*

**Example 2** In Figure 1, (c) has $OL_n^C = \Theta(\log n)$ while all of the other regions have $OL_n^C = \Theta(\sqrt{n})$.

Theorem 1 tells us that when $C$ does not have an upper-right-hand-corner then $\mathbf{E}\left(M_n^C\right) = \Theta(\sqrt{n})$. When $C$ does have such a corner then all that we know is that $\mathbf{E}\left(M_n^C\right) = O(\log n)$. To derive tighter bounds it is necessary to have better information about the tangents to the boundary of $C$ at the corner. We digress momentarily to introduce notation describing these tangents.

Let $C$ be a convex region. If $C$ has an upper-right-hand-corner $p$ then the boundary curve of $C$ as it leaves $p$ can be divided into two parts: one curve that goes down and the other that goes to the left. We define two functions $d(\alpha)$ and $l(\alpha)$ :

$$\begin{aligned} d(\alpha) &= \beta \text{ such that } (p.x - \beta, p.y - \alpha) \text{ is on the down curve,} \\ l(\alpha) &= \beta \text{ such that } (p.x - \alpha, p.y - \beta) \text{ is on the left curve.} \end{aligned}$$

See Figure 2. Although these functions are not defined for all real $\alpha$ the convexity of $C$ ensures that there is always some $\epsilon > 0$ such that both functions are well defined, convex and nondecreasing in $[0, \epsilon]$ with $d(0) = l(0) = 0$. Since the functions are convex their left and right derivatives exist everywhere in the interval except for the undefined left derivative at 0 and the undefined right one at $\epsilon$.

The down tangent to $C$ at $p$ is the tangent to the down curve at $p$. The slope of this tangent line is totally determined by the value of the right derivative $d'_+(0)$. If $d'_+(0) = 0$ the tangent line is vertical. Similarly, if $l'_+(0) = 0$ the left tangent line, the tangent to the left curve, is horizontal. This is illustrated in Figure 2.

The next two theorems discuss the behaviour of $\mathbf{E}\left(M_n^C\right)$ when $C$ has an upper-right-hand-corner .

**Theorem 2** *Let $C$ be a convex planar region with an upper-right-hand-corner $p$. If the down tangent at $p$ is not vertical and the left tangent at $p$ is not horizontal then $\mathbf{E}\left(M_n^C\right) = \Theta(1)$. Otherwise $\mathbf{E}\left(M_n^C\right) = \Omega(1)$.*

**Example 3** Figures 1 (b) and (f) have $\mathbf{E}\left(M_n^C\right) = \Theta(1)$; figures 1 (c), (d) and (h) have $\mathbf{E}\left(M_n^C\right) = \Omega(1)$.

We now present a tight lower bound for many of the cases in which the left tangent is horizontal and/or the down tangent is vertical.

**Theorem 3** *Let $C$ be a convex planar region with an upper-right-hand-corner $p$. Suppose further that at least one of the two tangents to $C$ at $p$ fulfills the following (Lipschitz-like) conditions:*

1. *The down tangent is not vertical and there are positive constants $\delta$ and $c$ such that $l(\alpha) \leq c\alpha^{1+\delta}$.*
2. *The left tangent is not horizontal and there are positive constants $\delta$ and $c$ such that $d(\alpha) \leq c\alpha^{1+\delta}$.*
3. *There are positive constants $\delta$ and $c$ such that $d(\alpha) \leq c\alpha^{1+\delta}$ and $l(\alpha) \leq c\alpha^{1+\delta}$.*

*Then $\mathbf{E}\left(M_n^C\right) = \Theta(\log n)$.*

Condition (1) forces the left tangent to be horizontal and condition (2) forces the down tangent to be vertical. The conditions of the theorem can be thought of as requiring not only the left (down) tangent to be horizontal (vertical) but the curve leaving $C$ itself to be "almost" horizontal (vertical) near $p$. These conditions might seem artificial but in practice are satisfied quite often. For example:

**Example 4** As an application of Theorem 3 we find that if $C$ is a convex *polygon* with an upper-right-hand-corner and a vertical down tangent and/or a horizontal left tangent at the corner then $\mathbf{E}\left(M_n^C\right) = \Theta(\log n)$, e.g. Figures 1 (c) and (h) have $\mathbf{E}\left(M_n^C\right) = \Theta(\log n)$.

Combining this with Theorems 1 and 2 we find that if $C$ is a convex polygon then $\mathbf{E}\left(M_n^C\right)$ can have only one of three possible behaviours: $\Theta(\sqrt{n})$, $\Theta(\log n)$, or $\Theta(1)$.

# 3 Dual Results

Let $C$ be a planar region. Choose $n$ points $p_1, \ldots, p_n$ I.I.D. from the uniform distribution over $C$. Let $M_n^C$ be the number of these points that are maximal. If $C$ is convex it is known that either $\mathbf{E}\left(M_n^C\right) = \Theta(\sqrt{n})$ or $\mathbf{E}\left(M_n^C\right) = O(\log n)$. We will show that, for general $C$, there is very little that can be said, a priori, about $\mathbf{E}\left(M_n^C\right)$. More specifically we will show that if $g$ is a member of a large class of monotonic functions then there is a region $C$ such that $\mathbf{E}\left(M_n^C\right) = \Theta(g(n))$. This class contains all functions with regular variation and (i) exponent less than 1 or (ii) exponent equal to 1 and $n/g(n) > \ln^\beta n$ for some $\beta > 1$. For example, all functions of the form $g(n) = n^\alpha$, $0 < \alpha < 1$, or $g(n) = \ln^\beta n$, $\beta \geq 0$ satisfy condition (i) while all functions of the form $g(n) = n \ln\ln n / \ln^\beta n$, $\beta > 1$ satisfy (ii). The class also contains nondecreasing functions like $g(n) = \ln^* n$. The results in this paper remain valid in higher dimensions.

## 3.1 Finding a planar region corresponding to "good" monotonic functions

**Definition 1** *A positive (not necessarily) monotone function $L$ defined on $(0,\infty)$* is slowly varying *at infinity if and only if, for all $x > 0$*

$$\lim_{t\to\infty} \frac{L(xt)}{L(t)} \to 1.$$

**Definition 2** *A function $U$ defined on $(0,\infty)$* is regularly varying *at infinity with exponent $\rho$ if and only if it is of the form $x^\rho L(x)$ where $L$ is slowly varying.*

As an example the function $\ln^2 n$ varies slowly at infinity so the function $\sqrt{n}\ln^2 n$ varies regularly at infinity with exponent $1/2$. Similarly, the function $1/\ln^2 n$ varies slowly at infinity so the function $n/\ln^2 n$ varies regularly at infinity with exponent 1.
We now state our main result.

**Theorem 4** *Let $g$ be a continuous, monotonically increasing almost everywhere differentiable function from $(0,\infty)$ onto itself. Furthermore, suppose that $g$ is regularly varying with exponent $\rho$ and either (i) $\rho < 1$ or (ii) $\rho = 1$ and $x/g(x) \geq \ln^\beta x$ for some $\beta > 1$. Then there is some planar region $C$ such that for $n$ points $p_1, \ldots, p_n$ chosen I.I.D. uniformly from $C$ the expected number of the points that are maximal is $\Theta(g(n))$ :*

$$\mathbf{E}\left(|\mathrm{MAX}\left(\{p_1, \ldots, p_n\}\right)|\right) = \Theta(g(n)).$$

Very large classes of functions $g$ satisfy the conditions of Theorem 4. Some examples:

1. $g(x) = x^\alpha$ where $\alpha < 1$.
2. More generally, $g(x) = x^\alpha e^{\ln^\beta x} \ln^\gamma x$ where $0 \leq \alpha < 1$, $0 \leq \beta < 1$ and $\gamma > 0$.
3. $g(x) = \ln^{(m)}(x)$ the $m$'th iterated logarithm: $\ln^{(0)}(x) = x$, $\ln^{(m+1)}(x) = \ln\left(\ln^{(m)}(x)\right)$.
4. $g(x) = \frac{x\ln\ln x}{\ln^\beta x}$ where $\beta > 1$.

Examples 1, 2, and 3 satisfy condition (i); example 4 satisfies condition (ii). The theorem therefore tells us that for each of these $g$-s there is some $C$ such that $\mathbf{E}\left(M_n^C\right) = \Theta(g(n))$.
A theorem due to Feller [5, page 281], giving the asymptotics of the truncated moments of regularly varying functions, allows us to show that if $g$ satisfies the conditions of Theorem 4 then

$$\sum_{i>g(n)} \frac{1}{g^{-1}(i)} = O\left(\frac{g(n)}{n}\right). \tag{1}$$

We demonstrate then the following theorem:

**Theorem 5** *Let $g$ be a continuous monotonically increasing function from $(0,\infty)$ into itself with $g(x) \leq x$ and $\lim_{x\to\infty} g(x) = \infty$. Use $g^{-1}$ to denote the functional inverse of $g$: $g(g^{-1}(x)) = g^{-1}(g(x)) = x$. Suppose that $g$ fulfills the following condition for all integers $n > 0$:*

$$\sum_{i>g(n)} \frac{1}{g^{-1}(i)} = O\left(\frac{g(n)}{n}\right). \tag{2}$$

*Then there exists a connected planar region $C$ such that, for $n$ points $p_1, \ldots, p_n$ chosen I.I.D. uniformly from $C$, the expected number of the points that are maximal is $\Theta(g(n))$ :*

$$\mathbf{E}\left(|\mathrm{MAX}\left(\{p_1, \ldots, p_n\}\right)|\right) = \Theta(g(n)).$$

The demonstration uses the following lemma:

**Lemma 1** *Fix $d > 0$ and let $T$ be the triangle (Figure 3) with vertices $(0,0)$, $(d,0)$ and $(2d,2d)$. Choose $n$ points $p_1, \ldots, p_n$ I.I.D. uniformly from $T$. Then*

$$\mathbf{E}\left(|\mathrm{MAX}\left(\{p_1, \ldots, p_n\}\right)|\right) \leq 2.$$

The demonstration proceeds in two parts, with the construction of two regions $C$ and $C'$.

*Part 1:* We construct an infinite sequence $C_i$ of smaller and smaller triangles of the type described by the lemma (Figure 4) and define $C = \bigcup_i C_i$. We show then that when $n$ points are chosen I.I.D. uniformly from $C = \bigcup_i C_i$ then $\mathbf{E}\left(M_n^C\right) = \Theta(g(n))$.

*Part 2:* We modify $C$ to yield a *connected* region $C'$ such that $\mathbf{E}\left(M_n^{C'}\right) = \Theta(g(n))$. It is this $C'$ that will satisfy the theorem. We need the following lemma:

**Lemma 2** *Let $d > 0$ and $d' \leq 4d$. Define $T$ to be, as in Lemma 1, the triangle with vertices $(0,0)$, $(d,0)$ and $(2d,2d)$. Let $R$ be the rectangle with vertices $(0,0)$, $(d,0)$, $(0,-d')$ and $(d,-d')$. If $p_1, \ldots, p_n$ are chosen I.I.D. uniformly from $T \cup R$ then*

$$\mathbf{E}(|\mathrm{MAX}(\{p_1, \ldots, p_n\})|) = O(1).$$

The new region $C'$ (Figure 5) will be the union of an infinite number of regions of the type defined by the lemma.

*Construction of $C'$ :*

1. Set $f_i = 1/\sqrt{g^{-1}(i)}$.
2. Set $x_1 = y_1 = 0$ and for $i > 1$ set $x_i = x_{i-1} + 2f_{i-1}$ and $y_i = y_{i-1} - 2f_i$.
3. Define $C'_1$ to be the triangle with vertices $(0,0), (f_1, 0)$ and $(2f_1, 2f_1)$. For $i > 1$ define $C'_i$ to be the triangle with vertices $(x_i+2f_i, y_i+2f_i), (x_i-2f_{i-1}, y_i-2f_{i-1})$ and $(x_i+f_i-f_{i-1}, y_i-2f_{i-1})$.
4. For $i > 0$ define $R_i$ to be the rectangle with vertices $(x_i, y_i), (x_i+f_i, y_i), (x_i+f_i, y_i-2f_i-2f_{i+1})$ and $(x_i, y_i - 2f_i - 2f_{i+1})$
5. Set $C' = \bigcup_i (C'_i \cup R_i)$.

### 3.2 Higher Moments

The regions $C$ constructed in this section were carefully tailored so that $\mathbf{E}\left(M_n^C\right) = \Theta(g(n))$ for given monotonically increasing functions $g$. There is a rather remarkable theorem due to Devroye [2] which, for $p > 1$, gives us the higher moments $\mathbf{E}\left(\left(M_n^C\right)^p\right)$. This theorem (more specifically the remark 3 following the theorem) states that if $\mathbf{E}\left(M_n^C\right) = \Theta(g(n))$ where $g$ is nondecreasing then $\mathbf{E}\left(\left(M_n^C\right)^p\right) = \Theta\left(\left(\mathbf{E}\left(M_n^C\right)\right)^p\right) = \Theta\left(g^p(n)\right)$. Thus we know all of the higher moments of $M_n^C$. As an example suppose $C$ was constructed so that $\mathbf{E}\left(M_n^C\right) = \Theta\left(n^{1/3}\right)$. Then $\mathbf{E}\left(\left(M_n^C\right)^p\right) = \Theta\left(n^{p/3}\right)$ for all $p \geq 1$.

## References


[1] J. L. Bentley, H. T. Kung, M. Schkolnick, and C. D. Thompson. On the average number of maxima in a set of vectors and applications. *Journal of the Association for Computing Machinery*, 25(4):536–543, October 1978.

[2] L. Devroye. Moment inequalities for random variables in computational geometry. *Computing*, 30:111–119, 1983.

[3] L. Devroye. *Lecture Notes on Bucket Algorithms*. Birkhauser Verlag, Boston, 1986.

[4] R. Dwyer. Kinder, gentler, average-case analysis for convex hulls and maximal vectors. *Sigact News*, 21(2):64–71, 1990.

[5] W. Feller. *An Introduction to Probability Theory and Its Applications*, volume II. John Wiley and sons, New York, 2 edition, 1971.

[6] M. Golin. Notes on maxima in non-convex regions. Rapport de recherche 1535, Institut National de Recherche en Informatique et en Automatique, 1991. Extended Version.

[7] A. Renyi and R. Sulanke. Über die konvexe Hulle von $n$ zufallig gewahlten Punkten, I. *Z. Wahrscheinlichkeitstheorie*, 2(75–84), 1963.

[8] A. Renyi and R. Sulanke. Über die konvexe Hulle von $n$ zufallig gewahlten Punkten, II. *Z. Wahrscheinlichkeitstheorie*, 3(138–147), 1964.

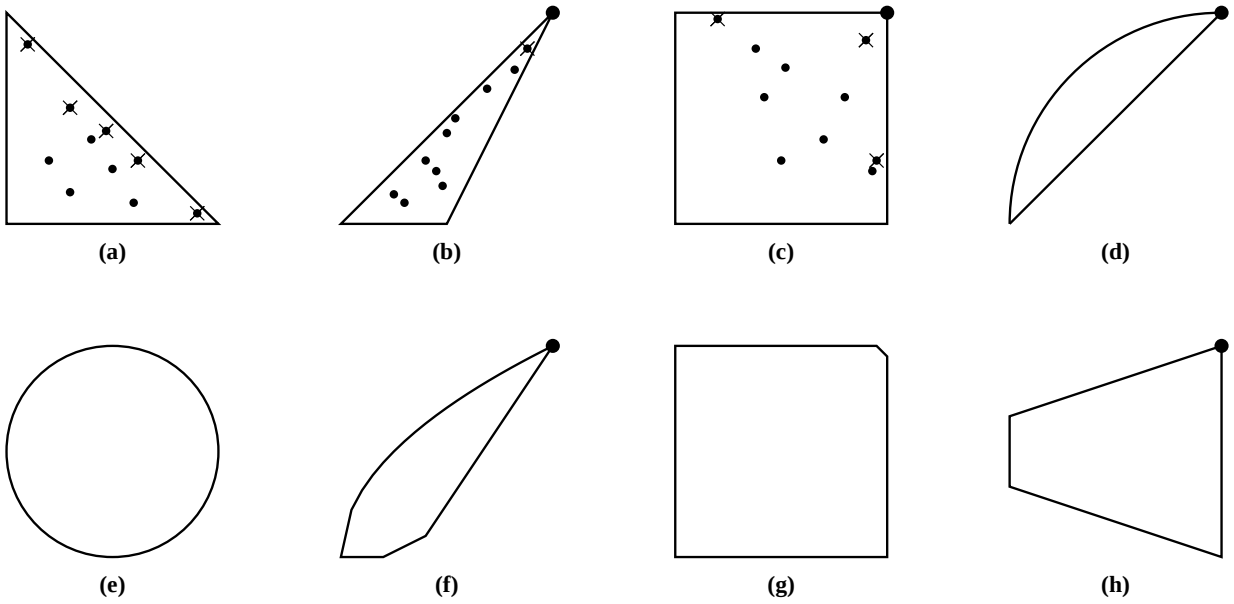



Figure 1: Figures (a), (b), and (c) each contain 10 points the maxima of which are marked by x-s; (a) contains 5 maximal points, (b) 1 maximal point and (c) 3 maximal points. Figures (b), (c), (d), (f) and (h) all have upper-right-hand-corners (marked with a large point); the other figures don't have a corner. Figures (c) and (d) have horizontal left tangents; figures (c) and (h) have vertical down ones.

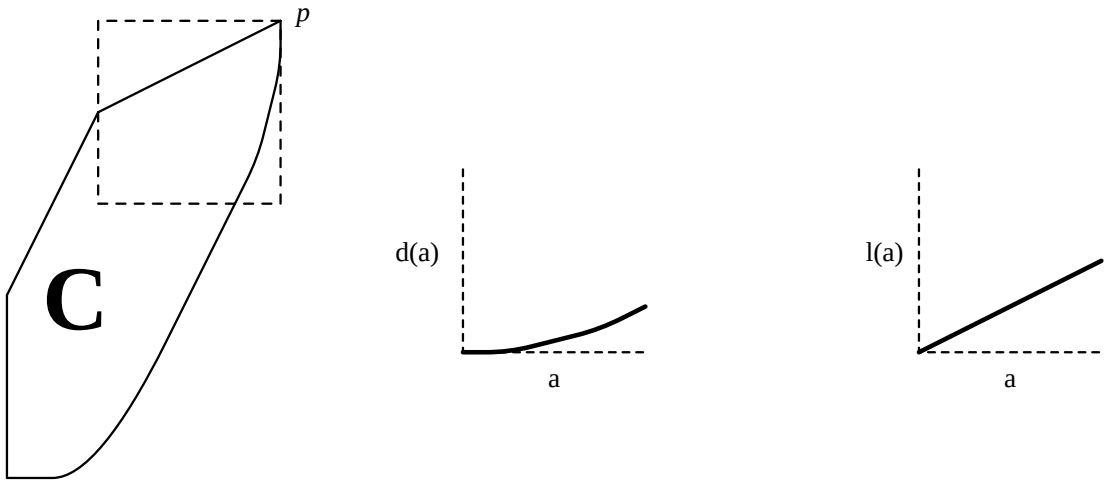



Figure 2: $C$ has an upper-right-hand-corner $p$ with a vertical down tangent at $p$ and a non-horizontal left one. The middle figure portrays $d(\alpha)$, the displacement of the down boundary curve from the vertical line through $p$ and the rightmost figure portrays $l(\alpha)$, the displacement between the left boundary curve and the horizontal line through $p$.

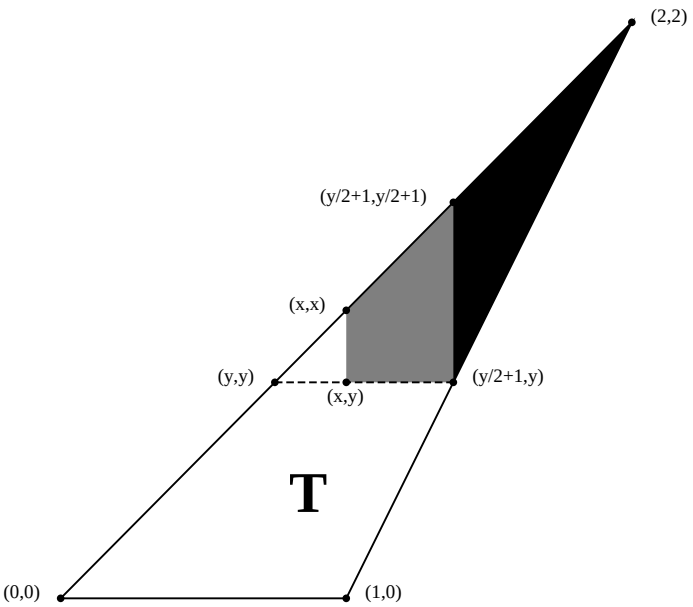



Figure 3: $R(x,y)$ is the region containing all points in $T$ that dominate $(x,y)$. In the diagram $R(x,y)$ is the union of the two shaded regions. The darker of the two shaded regions (the small triangle) has area $\frac{1}{2}[y/2+1-y][2-(y/2+1)] = \frac{1}{2}[1-y/2]^2$ so Area$(R(x,y)) \geq \frac{1}{2}[1-y/2]^2$.

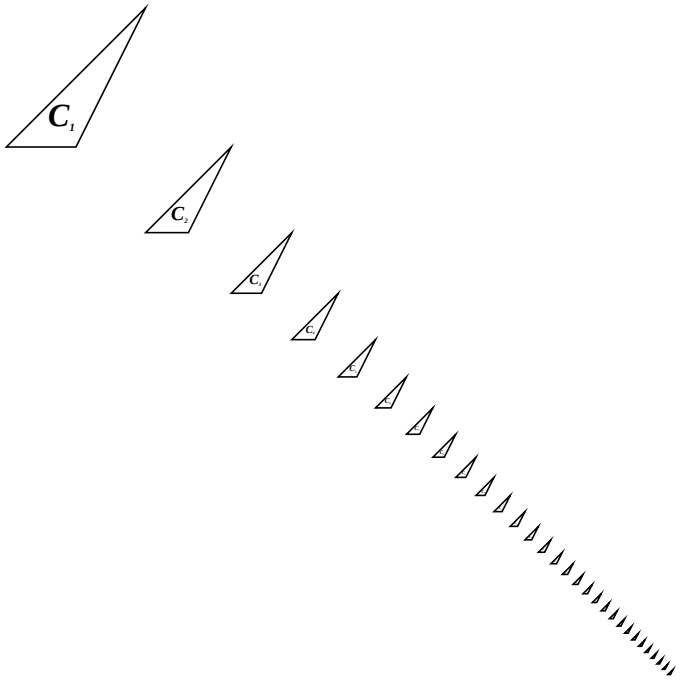



Figure 4: The region $C = \cup_i C_i$ when $g(x) = x^{5/12}$, $g^{-1}(x) = x^{12/5}$ and $f_i = i^{-6/5}$. Note that points in different triangles are incomparable; if $p \in C_i$ and $q \in C_j$ where $i \neq j$ then $p$ and $q$ are incomparable.

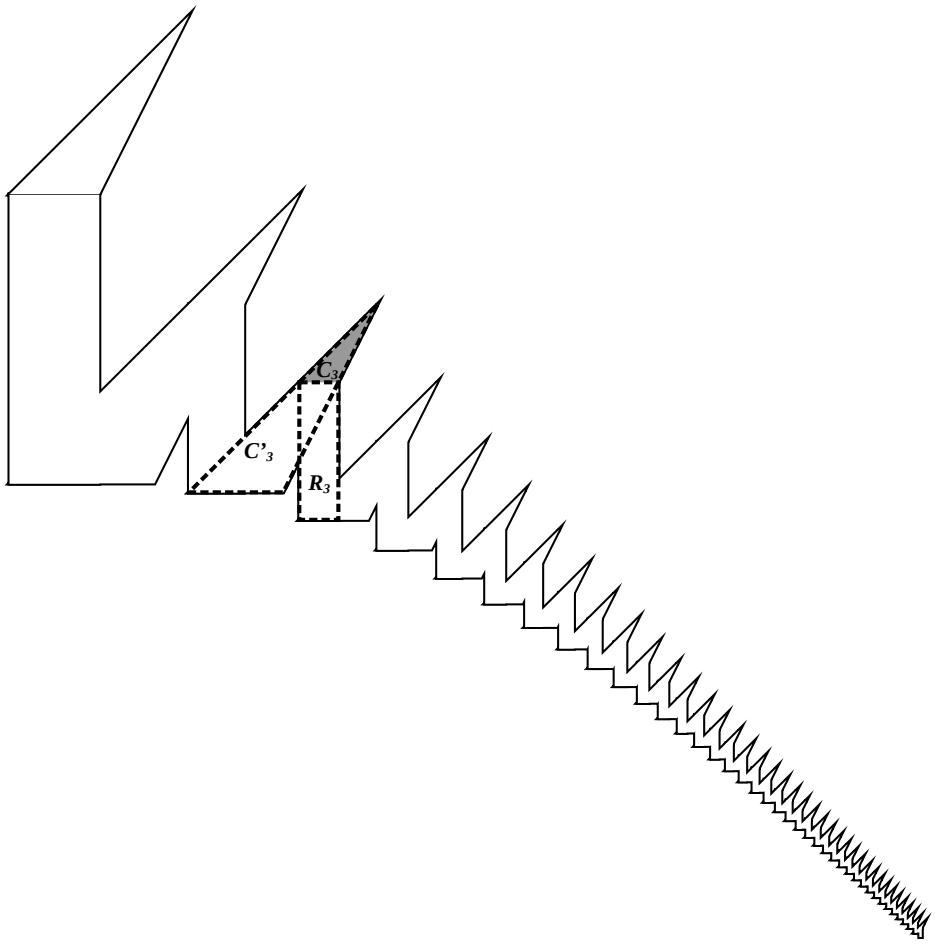



Figure 5: The region $C' = \bigcup_i (C'_i \cup R_i)$ when $g(x) = x^{5/12}$. We have emphasized $C'_3$ and $R_3$ by giving them a dashed boundary. The shaded region is the triangle $C_3$ of the preceding figure. Note how $C_3 \subseteq C'_3$.

# 4

# Fourier Transforms over Semi-simple Algebras

François Bergeron
UQAM, Montréal

[summary by Dominique Gouyou-Beauchamps]

## 1 Introduction

Given a finite group $G$, we study some aspects of probabilistic algorithms of the form:

$r := 1$

**repeat**

choose $g \in G$ with probability $p(g)$

$r := r.g$

**until** the probability distribution of $r$ is close to uniform

where $p : G \longrightarrow [0,1]$ is a probability distribution on $G$. We are also interested in some questions such as the explicit computation of the probability of obtaining some element $g \in G$ after $n$ iterations of the loop. This study is clearly equivalent to the computation of successive powers, in the group algebra $\mathcal{A}(G)$ of $G$, of $\alpha = \sum_{g \in G} p(g)g$. However these powers are often hard to calculate in a nice closed form.

**Example 1**: We have an $n$-tuple of bits $w \in \{0,1\}^n$. At time $t = 1, 2, 3, ...$, we randomly choose one bit of $w$ and we change the value of this bit ($0 \leftrightarrow 1$). At time $t$, what is the probability that we obtain a given $n$-tuple $w_0 \in \{0,1\}^n$?

**Example 2**: We consider $G = \mathcal{S}_n$, the symmetric group, and we compute powers of $\frac{1}{2^n}\sum_{j=0}^{n} 1\,2...j \sqcup\!\sqcup (j+1)(j+2)...n$ where $\sqcup\!\sqcup$ denotes the shuffle product. This problem has been considered by Diaconis in [1, 2].

**Example 3**: We can also consider powers of $\frac{1}{\binom{n}{2}}\sum_{i \neq j}(i,j)$ where $(i,j)$ denotes the transposition that exchanges $i$ and $j$.

All these formulas can be considered to give, in explicit form, a Fourier transform such as defined below.

Let $\mathcal{A}$ be a semi-simple commutative algebra (i.e. having a basis of orthogonal idempotents), and let $B = v_1, v_2, ..., v_n$ be some fixed (linear) basis for the subalgebra $\mathcal{B}$ of $\mathcal{A}$, spanned by the complete set of primitive idempotents $e_1, e_2, ..., e_n$ of $\mathcal{A}$. Recall that these idempotents are such that

$$e_k e_j = \begin{cases} e_k & \text{if } k = j \\ 0 & \text{otherwise} \end{cases} \qquad \text{and} \qquad \sum_{k=1}^{n} e_k = 1.$$

Moreover, none of them can be written as the sum of two orthogonal idempotents. The *Fourier transform* $\hat{f}$ (with respect to B) of $f = \sum_k f_k v_k \in \mathcal{B}$ is defined to be the vector $(\hat{f}_1, \hat{f}_2, ..., \hat{f}_n)$ of coordinates of $f$ in this canonical basis $(e_k)_{1 \leq k \leq n}$

$$f = \sum_{k=1}^{n} \hat{f}_k e_k.$$

And the powers of $f$ can be written

$$f^N = \sum_{k=1}^{n} \hat{f}_k^N e_k.$$

## 2 Example 1

Let $\mathcal{B}$ be the center $C(G)$ of the group algebra of a finite group $G$. The basis $B$ is chosen to be the set of conjugacy classes in $G$

$$c_\rho = \sum_{g \in c(\rho)} g,$$

where $c(\rho) = \{h^{-1}\rho h \mid h \in G\}$ and $\rho \in G$.
Then the canonical idempotents are essentially given by the characters $\chi_\rho$ of irreducible representations $\rho$ of $G$, considered as elements of the group algebra

$$e_\rho = \frac{\chi_\rho(1)}{|G|} \sum_{g \in G} \chi_\rho(g^{-1}) g.$$

If $G = \mathcal{S}_n$, a conjugacy class is a partition of the integer $n$ because two permutations are in the same class if and only if they have the same cycle decomposition.
If $G = \langle x \rangle$, the cyclic group of order $n$ generated by $x$ ($x^n \equiv 1$), since the group is abelian, any element of the group algebra of $\langle x \rangle$

$$f(x) = \sum_{k=0}^{n-1} f_k x^k,$$

is an element of the center $C(\langle x \rangle)$. The group algebra can be seen as $\mathcal{A} = \mathbb{C}[x]/\langle x^n - 1 \rangle$. Moreover the irreducible characters give, in this case, the following idempotents

$$e_j = \frac{1}{n} \sum_{k=0}^{n-1} e^{-2ikj\pi/n} x^k,$$

for $0 \leq j \leq n-1$. One concludes that

$$f(x) = \sum_{j=0}^{n-1} \hat{f}_j e_j,$$

where

$$\hat{f}_j = \frac{1}{n} \sum_{k=0}^{n-1} f^k e^{-2i(n-k)j\pi/n}$$

$$= \frac{1}{n} f(q^j).$$

This is the traditional definition of the discrete Fourier Transform.

# 3 Example 2

The semi-simple algebras considered in this example are subalgebras of the group algebra of finite Coxeter groups. As a guiding example, let us consider the semi-simple subalgebra $\Gamma[A_{n-1}] = \Gamma[\mathcal{S}_n]$ of the symmetric group spanned by the linearly independent descent classes

$$D_k = \sum_{d(\sigma)=k} \sigma,$$

where $0 \leq k \leq n-1$ and $d(\sigma) = \mathrm{Card}\{1 \leq i \leq n-1 \mid \sigma(i) > \sigma(i+1)\}$ (cf [5]). Now, in [4] A. Garsia gives a beautiful explicit formula

$$\sum_{k=1}^{n} t^k e_k = \frac{1}{n!} \sum_{k=0}^{n-1} (t-k)^{(n)} D_k,$$

relating the basis $D_k$ and the canonical idempotents $e_k$ of $\Gamma(\mathcal{S}_n)$. Here, $(t)^{(n)}$ stands for the rising factorial

$$(t)^{(n)} = t(t+1)(t+2)...(t+n-1).$$

This formula is closely related to the shuffle algebra and to a problem considered by Diaconis in [1, 2]. It is used in [2] to study the number of shuffles needed in order to really mix a deck of $n$ cards. If we denote

$$\hat{\alpha} = \widehat{\sum_g \alpha_g g} = \sum_g \alpha_g g^{-1},$$

then $\widehat{D_{\leq i}} = w_1 \sqcup\!\sqcup w_2 \sqcup\!\sqcup ... \sqcup\!\sqcup w_i$ if $w_1 w_2 ... w_i = 1\,2...n$.
Example: $\sigma = 34681257$, $d(\sigma) = 1$, $\sigma^{-1} = 56127384 \in 1234 \sqcup\!\sqcup 5678$.
Applying Garsia's formula with $t = 2$ gives

$$\sum_{k=1}^{n} \frac{2^k}{2^n} \widehat{e_k} = \frac{1}{2^n} \sum_{k=0}^{n-1} 1\,2...k \sqcup\!\sqcup (k+1)...n \ .$$

The idempotent $e_n$ is dominating in this formula because its coefficient is the greatest. But $e_n$ is equal to $\sum_{\sigma \in \mathcal{S}_n} \frac{1}{n!}\sigma$ which is nothing else than the uniform distribution.
For $t \geq 1$, $\frac{1}{t^n} \sum_{k=1}^{n} t^k e_k = \sum_{k=0}^{n-1} \frac{(t-k)^{(n)}}{n! t^n} D_k$ is a probability distribution on $\mathcal{S}_n$. It follows immediately from the orthogonality of the $e_k$'s that

$$\left( \sum_{k=0}^{n-1} \frac{(t-k)^{(n)}}{n! t^n} D_k \right)^j = \left( \frac{1}{t^n} \sum_{k=1}^{n} t^k e_k \right)^j = \frac{1}{t^{jn}} \sum_{k=1}^{n} t^{jk} e_k = \sum_{k=0}^{n-1} \frac{(t^j-k)^{(n)}}{n! t^{jn}} D_k.$$

In order to generalise this last computation, we extend Garsia's formula using an umbral argument. Thus we obtain

$$\sum_{k=1}^{n} t_k e_k = \sum_{k=0}^{n-1} \left( \sum_{j=1}^{n} \Psi_n(k,j) t_j \right) D_k,$$

where the $\Psi_n(k,j)$ are the coefficients appearing in the expression of the polynomial

$$\frac{1}{n!}(t-k)^{(n)} = \sum_{j=1}^{n} \Psi_n(k,j) t^j.$$

Hence if $\Phi_n$ stands for the inverse of the matrix $\Psi_n$, then the Fourier transform, with respect to the basis $(D_k)_{0 \leq k \leq n-1}$, is

$$\widehat{s^k} = \sum_{j=1}^{n} \Phi_n(k,j) s^j.$$

Thus $\Phi_n$ is the matrix for the Fourier transform and $\Psi_n$ that for the inverse Fourier transform. For example

$$\Phi_1 = [1] \qquad \Phi_2 = \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} \qquad \Phi_3 = \begin{bmatrix} 1 & -2 & 1 \\ 1 & 0 & -1 \\ 1 & 4 & 1 \end{bmatrix} \qquad \Phi_4 = \begin{bmatrix} 1 & -3 & 3 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & 3 & -3 & -1 \\ 1 & 11 & 11 & 1 \end{bmatrix}$$

$$\Phi_5 = \begin{bmatrix} 1 & -4 & 6 & -4 & 1 \\ 1 & -2 & 0 & 2 & -1 \\ 1 & 2 & -6 & 2 & 1 \\ 1 & 10 & 0 & -10 & -1 \\ 1 & 26 & 66 & 26 & 1 \end{bmatrix} \qquad \Phi_6 = \begin{bmatrix} 1 & -5 & 10 & -10 & 5 & -1 \\ 1 & -3 & 2 & 2 & -3 & 1 \\ 1 & 1 & -8 & 8 & -1 & -1 \\ 1 & 9 & -10 & -10 & 9 & 1 \\ 1 & 25 & 40 & -40 & -25 & -1 \\ 1 & 57 & 302 & 302 & 57 & 1 \end{bmatrix}.$$

The last row of these matrices is readily seen to be given by the coefficients of the Eulerian polynomials

$$\mathbf{A}_n(x) = \sum_{\sigma \in \mathcal{S}_n} x^{d(\sigma)},$$

hence

$$\Phi_n(n,k) = \mathrm{Card}\{\sigma \in \mathcal{S}_n \mid d(\sigma) = k\}.$$

In general the entries of $j^{th}$ row of $\Phi_n$ are the coefficients of $(1-x)^{(n-j)}\mathbf{A}_j(x)$.

## 4 Example 3

Similar consideration can be made in the context of the group algebra of the hyperoctahedral group ($\mathcal{B}_n$ in the Coxeter's classification), if one considers the semi-simple subalgebra $\Gamma[\mathcal{B}_n]$ of the symmetric group spanned by the linearly independent descent classes

$$D_k = \sum_{d(\sigma)=k} \sigma,$$

where $0 \leq k \leq n$ and $d(\sigma) = \mathrm{Card}\{1 \leq i \leq n-1 \mid \sigma(i) > \sigma(i+1)\}$.
Recall that elements of $\mathcal{B}_n$ are signed permutations and that for sake of convenience one can set $\sigma(0) = 0$. It was shown in [3] that there is in this context a Garsia like formula

$$\sum_{k=0}^{n} t^k e_k = \frac{1}{2^n n!} \sum_{k=0}^{n} (t-2k)^{((n))} D_k,$$

relating the basis $D_k$ and the canonical idempotents $e_k$ of $\Gamma(\mathcal{B}_n)$. Here, $(t)^{((n))}$ stands for the double rising factorial

$$(t)^{((n))} = (t+1)(t+3)...(t+2n-1).$$

As in the previous case, the last row of the matrices for the Fourier transform is readily seen to be given by the coefficients of the hyperoctahedral descent polynomials

$$\mathbf{B}_n(x) = \sum_{\sigma \in \mathcal{B}_n} x^{d(\sigma)},$$

and the entries of $j^{th}$ row of $\Phi_n$ are the coefficients of $(1-x)^{(n-j)}\mathbf{B}_j(x)$. It is easy to verify that the polynomials $\mathbf{B}_n(x)$ satisfy the following recurrence

$$\mathbf{B}_{n+1}(x) = (1+x)\mathbf{B}_n(x) + 2xn\mathbf{B}_n(x) + (2x-2x^2)\tfrac{d}{dx}\mathbf{B}_n(x),$$

hence that their exponential generating function is

$$\sum_{n\geq 0} \mathbf{B}_n(x)\tfrac{u^n}{n!} = \tfrac{(1-x)e^{u(1-x)}}{1-xe^{2u(1-x)}}.$$

# References


[1] D. Aldous and P. Diaconis. Shuffling cards ans stopping times. *American Mathematical Monthly*, 93:333–348, 1986.

[2] D. Bayer and P. Diaconis. Trailing the dovetail shuffle to its lair. Technical Report 2, Dept. of Statistics, Stanford University, 1990.

[3] F. Bergeron and N. Bergeron. Orthogonal idempotents in the descent algebra of $B_n$. *Journal of Pure and Applied Algebra*, accepted in 1991.

[4] A. Garsia. Combinatorics of the free Lie algebra and the symmetric group. In P. H. Rabinowitz and E. Zehnder, editors, *Research papers published in honor of Jurgen Moser's* $60^{th}$ *birthday*. Academic Press, 1990.

[5] A. Garsia and C. Reutenauer. A decomposition of Solomon's descent algebras. *Advances in Mathematics*, 77(2):189–262, 1989.

# 5

# Suites 2-régulières et séries rationnelles

Philippe Dumas
INRIA, Rocquencourt

Les suites 2-régulières apparaissent naturellement dans les problèmes de comptage liés à l'écriture binaire des entiers ou dans l'étude d'algorithmes du type "diviser pour régner". L'exemple le plus classique en est la suite de Thue-Morse qui donne la parité du nombre de 1 dans le développement binaire.
Le but de l'exposé est de montrer et d'illustrer le lien profond qui existe entre les séries rationnelles, au sens de la théorie des langages, et les suites 2-régulières. Celles-ci ne sont qu'une traduction de celles-là et la riche théorie des séries rationnelles fournit des résultats sur la représentation linéaire, la forme des récurrences ou le comportement des termes des suites 2-régulières.

## Références

[1] J.-P. Allouche and J. Shallit. The ring of $k$-regular sequences. *Theoretical Computer Science*, 98:163–197, 1992.

[2] Ph. Dumas. *Récurrences Mahlériennes, suites automatiques, et études asymptotiques.* Doctorat de mathématiques, Université de Bordeaux I, 1992. In preparation.

# Part II

# Generating Functions and Symbolic Computation

# 6

# Approximations de séries génératrices

Simon Plouffe
UQAM, Montréal

[résumé par Paul Zimmermann]

Quelle est la suite logique de $1, 2, 4, 7$ ? L'exposé montre que pour un certain nombre de suites, on peut trouver automatiquement une suite logique, en utilisant le calcul formel. L'idée est d'essayer de trouver une forme explicite pour la série génératrice associée aux termes de la suite, à partir des premiers. Lorsqu'une telle forme existe, et qu'elle correspond effectivement à la suite d'entiers (ce qui n'est pas toujours facile à vérifier), un simple développement de Taylor donne n'importe quel terme. On peut même obtenir un développement asymptotique du $n$-ième terme en utilisant un module d'analyse asymptotique comme le programme `equivalent` de B. Salvy [4].

## 1 Le livre de Sloane : *A Handbook of Integer Sequences*

Pour trouver les prochains termes après $1, 2, 4, 7$, on peut remarquer que le second terme égale le premier augmenté de 1, le troisième égale le second augmenté de 2, le quatrième le troisième augmenté de 3. La suite logique est donc $7+4 = 11$, puis $11+5 = 16, 16+6 = 22$. Une autre méthode est de consulter une table de suites, en l'occurrence le livre *A Handbook of Integer Sequences* publié en 1973 par Sloane [5]. Ce livre contient une table de 2372 entrées, chaque entrée étant constituée d'un numéro d'index, des premiers termes de la suite, d'une définition de la suite et d'une référence. La suite $1, 2, 4, 7, 11, 16, 22$ y est répertoriée sous le numéro 391. On y découvre que ces nombres sont appelés "nombres polygonaux centraux", valent $n(n-1)/2+1$, et sont aussi le nombre maximal de parts que l'on peut obtenir en coupant un gâteau $n-1$ fois.
Mais d'autres suites commençant par $1, 2, 4, 7$ existent : il y en a en tout 25 répertoriées dans le livre de Sloane (numéros 388 à 412). Ainsi il y a 24 autres suites possibles à $1, 2, 4, 7$ : par exemple $1, 2, 4, 7, 8$ (entiers qui ont un nombre impair de 1 dans leur écriture binaire) ou $1, 2, 4, 7, 12$ (nombres de Fibonacci moins un).
Depuis la parution de son livre en 1973, Sloane a reçu un courrier volumineux (près d'un mètre cube de lettres) lui indiquant de nouvelles suites; et depuis quelque temps, il a entrepris avec Simon Plouffe la réalisation d'une seconde édition du *Handbook of Integer Sequences*. Cette nouvelle édition comprendra environ 5000 suites, 92000 termes, 6200 références, et des formules.

## 2 La fonction `'convert/ratpoly'` de Maple

Il est en effet intéressant d'avoir un moyen simple de générer les termes d'une suite. Cela peut être une formule de récurrence, ou une série génératrice. Par exemple, les nombres polygonaux centraux sont les coefficients du développement de Taylor à l'origine de $(1 - z + z^2)/(1 - z)^3$ :

```
> f:=(1-z+z^2)/(1-z)^3:
> series(f,z,7);
                          2      3       4       5       6      7
             1 + 2 z + 4 z  + 7 z  + 11 z  + 16 z  + 22 z  + O(z )

> factor(convert(",ratpoly));
                                        2
                                  1 - z + z
                                - ----------
                                          3
                                   (z - 1)
```

Ainsi, la fonction `'convert/ratpoly'` permet de retrouver la fraction rationnelle $f$ à partir des premiers termes, et est en quelque sorte l'inverse de `series` pour les fractions rationnelles.
Autre exemple : les nombres merveilleux de Demlo (suite 2339 de [5]) sont 1, $11^2 = 121$, $111^2 = 12321$, 1234321, 123454321, 12345654321, 1234567654321, ... Avec ces seuls sept termes, la fonction `'convert/ratpoly'` de Maple trouve la série $(1 + 10z)/((1 - z)(1 - 10z)(1 - 100z))$. Lorsqu'on trouve une forme rationnelle $P(x)/Q(x)$, le polynôme $P$ traduit les conditions initiales et $Q$ la récurrence.
La fonction `'convert/ratpoly'` de Maple, qui a été écrite par K. Geddes, utilise la méthode de Cabay-Choi pour calculer les approximants de Padé [2]. C'est l'un des meilleurs algorithmes connus, et le résultat est réduit au maximum. Cet algorithme donne toujours une approximation sous forme de fraction rationnelle, dont la somme des degrés du numérateur et du dénominateur est au plus égale au nombre de termes donnés. Ainsi, pour s'assurer que l'on a obtenu la "bonne" forme, on recommence avec quelques termes supplémentaires et on vérifie que le résultat est le même.
Sur 4568 suites, la seule utilisation de `'convert/ratpoly'` a permis de trouver 600 séries génératrices rationnelles! Par exemple, les nombres de Delaunay $(1, 7, 25, 63, 129, \ldots$, suite 1844) ont pour série génératrice $(1+x)^3/(1-x)^4$, les permanents des matrices $(0, 1)$ cycliques $(1, 24, 44, 80, 144, 264, \ldots$, suite 2232) ont pour série $4(6 - x - 2x^2 - 4x^3)/((1 - x)(1 - x - x^2 - x^3))$, le nombre de façons de remplir une boîte avec des dominos $(1, 2, 2, 4, 5, 9, \ldots$, suite 117) a pour série $(1 + x - 2x^2 - x^3 - x^4 - x^5)/((1 - x - x^2)(1 - x^2 - x^4))$.
Le principal avantage de la représentation par séries génératrices est que cela permet de générer facilement des termes supplémentaires. D'autre part, la complexité de la formule permet d'apprécier la "généricité" de la suite (en effet, si les coefficients sont petits, alors la suite s'exprime simplement en tant que série génératrice). Cela permet de regrouper les suites par classes de formules (rationnelles, algébriques, etc), de déduire une formule générale pour une famille de suites semblables (par exemples les nombres de Delaunay).
Le principal problème est justement de trouver la forme de la série génératrice associée à une suite, lorsqu'elle existe. La fonction `'convert/ratpoly'` de Maple reconnaît les fractions rationnelles, mais cela ne représente que 600 suites sur 4568, soit seulement 13%.

## 3 Extension à d'autres classes de formules

Lorsque `'convert/ratpoly'(f)` ne donne pas de forme simple, l'idée (due à F. Bergeron) est alors de regarder si $\log(f)$, $\exp(f)$, la dérivée de $f$, sa dérivée logarithmique, son inverse fonctionnel, et d'autres transformées de $f$ ont une forme rationnelle. On "récupère" ainsi les formules du type

$\log(P/Q)$, $\exp(P/Q)$, $\sqrt{P/Q}$, $\int P/Q$ et leurs combinaisons. En fait, il est apparu à l'expérience que trois transformations attrapent à elles seules la plupart des formules : la dérivée, la dérivée logarithmique et l'inverse pour la substitution.
Ces transformations permettent de trouver un plus grand nombre de séries : près de 1000 sur 4568, soit environ 22%. Par exemple le nombre de nuées de $n$ points $(1, 3, 12, 70, 465, \ldots$, suite 1181) a pour série $\exp(-z/2 - z^2/4)/(1-z)^{1/2}$, la seconde colonne des nombres de Stirling de première espèce $(1, 3, 11, 50, 274, \ldots$, suite 1165) a pour série génératrice exponentielle $(1 - \log(1-z))/(1-z)^2$, le nombre de permutations de $n$ éléments sans cycle de longueur 3 (suite 496) a pour série $\exp(-z^3/3)/(1-z)$.
S'il est facile d'utiliser la fonction '`convert/ratpoly`' de Maple pour trouver une formule, il n'est pas toujours aussi simple de prouver que cette formule correspond réellement à la suite en question. Prolonger la suite n'est pas évident dans certains cas. Par exemple, la suite correspondant au nombre de graphes enracinés de genre 2 sur $n$ points, dont les termes connus sont 21, 483, 6468, 66066, 570570, 4390386, 31039008, 205633428, 1293938646, 7808250450, 45510945480, semble avoir comme série génératrice $21(1+z)/(1-4z)^{11/2}$, mais ce n'est qu'une conjecture.

# 4 Remarques finales

La fonction '`convert/ratpoly`' n'est pas limitée aux séries à une variable. Par exemple, elle permet de trouver la série bivariée des polynômes de Tchébychev :

```
> 1 + x*t + (2*x^2-1)*t^2 + (4*x^3-3*x)*t^3 + (8*x^4-8*x^2+1)*t^4:
> convert(series(",t),ratpoly);
                                    - t x + 1
                                  ---------------
                                             2
                                  - 2 t x + t  + 1
```

Au lieu de chercher une formule close pour la série génératrice associée à une suite, on peut chercher une équation qu'elle vérifie (équation algébrique ou différentielle). B. Salvy a écrit un programme Maple qui cherche soit une équation algébrique à coefficients polynomiaux, soit une équation différentielle linéaire à coefficients polynomiaux également, par la méthode des coefficients indéterminés. Ce programme trouve par exemple la forme suivante pour les graphes polyédraux enracinés :

```
                                3/2                 2      3      4       5
                    (1 + x) ((- 4 x + 1)    - 1 + 6 x - 6 x  - 4 x  - 6 x ) + 4 x
                1/2 ------------------------------------------------------------
                                      5        3
                                     x  (x + 2)  (1 + x)
```

et pour les dénominateurs des convergents du nombre $e$ (suite 1240) :

```
                                                  1/2
                                  exp(1/2 - 1/2 (1 - 4 x)   )
                                  ---------------------------
                                                1/2
                                       (1 - 4 x)
```

En conclusion, il a été réalisé par Simon Plouffe et François Bergeron [1] un programme Maple qui, à partir des premiers coefficients d'une suite, cherche une forme simple pour la série génératrice

ordinaire (ou exponentielle) associée, en utilisant éventuellement des transformations élémentaires sur la suite. Ce programme trouve une formule pour environ 22% des suites de la nouvelle édition, soit 1000 suites sur 4568. Parmi les suites qui restent, le programme `guessgf` de B. Salvy détecte 20 suites algébriques et 194 suites holonomes. Enfin, en combinant le programme de Simon Plouffe et le programme `equivalent` de Bruno Salvy, on obtient une fonction Maple qui donne le développement asymptotique d'une suite à partir des premiers coefficients :

```
> Asympt(21,483,6468,66066,570570,4390386);
```

$$\left(8/9\ \frac{n^{9/2}\ 4^n}{\pi^{1/2}}\right)$$

**Pour envoyer de nouvelles suites :** Les adresses électroniques de N. J. A. Sloane et Simon Plouffe sont respectivement `njas@research.att.com` et `plouffe@lacim.uqam.ca`.

## Références

[1] F. Bergeron and S. Plouffe. Computing the Generating Function of a Series given its First Terms. *Journal of experimental mathematics*, 1992. Existe aussi en rapport technique numéro 164 du Département de Mathématiques et d'Informatique de l'Université du Québec à Montréal.

[2] S. Cabay and D.-K. Choi. Algebraic Computations of Scaled Padé Fractions. *SIAM Journal on Computing*, 15(1):243–270, February 1986.

[3] S. Plouffe. Approximations de séries génératrices et quelques conjectures. Rapport de recherche 61, Laboratoire Bordelais de Recherche en Informatique, 1992. Mémoire présenté à l'Université du Québec à Montréal comme exigence partielle de la maîtrise en Mathématiques.

[4] B. Salvy. *Asymptotique automatique et fonctions génératrices.* Thèse de doctorat, École Polytechnique, 1991.

[5] N. J. A. Sloane. *A Handbook of Integer Sequences.* Academic Press, 1973.

# 7

# Autour des nombres et fonctions algébriques en Maple

Marc Rybowicz
Université de Limoges

[résumé par François Morain]

La simplification d'expressions algébriques contenant des radicaux est une tâche assez ardue, encore mal algorithmisée. Par exemple, en MAPLE, il est facile de trouver les racines d'un polynôme du troisième degré en utilisant les formules de Cardan :

```
    |\^/|      MAPLE V
._|\|   |/|_.  Copyright (c) 1981-1990 by the University of Waterloo.
 \  MAPLE  /   All rights reserved.  MAPLE is a registered trademark of
 <____ ____>   Waterloo Maple Software.
      |        Type ? for help.
> p:=x^3+x+1;
                                   3
                          p := x  + x + 1

> solve(");
                                                    1/2
          %2 + %1, - 1/2 %2 - 1/2 %1 + 1/2 3    (%2 - %1) I,

                                        1/2
              - 1/2 %2 - 1/2 %1 - 1/2 3    (%2 - %1) I

                                    1/2  1/2 1/3
%1 :=                 (- 1/2 - 1/18 31    3   )

                                    1/2  1/2 1/3
%2 :=                 (- 1/2 + 1/18 31    3   )
```

Par contre, si l'on veut vérifier que ce sont vraiment les racines de `p`, il faut substituer ces valeurs dans `p` et simplifier les expressions obtenues. Dans l'état actuel (MAPLE V), il est impossible de montrer que les expressions obtenues sont nulles, d'une façon algébrique. Bien sûr, une évaluation flottante est possible et confirme la validité des formules.
Cet exposé se propose de passer en revue quelques méthodes de simplification de radicaux et de préciser l'implantation de celles-ci dans la procédure `radnormal` de MAPLE, réalisée par l'auteur.

## 1 Présentation du problème

Soit $k$ un corps de nombres et $\alpha_1$, $\alpha_2$, ..., $\alpha_n$ des éléments de $k$, $r_1$, $r_2$, ..., $r_n$ des entiers naturels. On pose

$$K = k(\alpha_1^{1/r_1}, \alpha_2^{1/r_2}, \ldots, \alpha_n^{1/r_n}).$$

Le problème qui se pose est de déterminer le degré de l'extension $[K : k]$ et/ou de calculer une base de $K/k$.
Les premiers à aborder ce problème ont été Caviness et Fateman [2], puis Zippel [3]. Borodin *et al.* ont étudié la simplification d'expressions faisant intervenir des racines carrées [1].

# 2 Quelques méthodes de résolution

## 2.1 Une méthode brutale

Cette méthode consiste à factoriser $X_1^{r1} - \alpha_1$, à choisir un facteur $P_1$, puis à factoriser $X_2^{r2} - \alpha_2$ sur $k[X_1]/(P_1)$, etc. A la fin du processus, on a construit

$$K \simeq k[X_1, X_2, \ldots, X_n]/(P_1 P_2 \ldots P_n) = \{X_1^{e_1} X_2^{e_2} \ldots X_n^{e_n}, 0 \leq e_i < \deg(P_i)\}$$

Considérons par exemple le cas :

$$k = \mathbb{Q}(\sqrt{2}, \sqrt{3}, \sqrt{-5}),$$
$$\alpha_1 = 1 + \sqrt{-5}, \quad \alpha_2 = 1 - \sqrt{-5},$$
$$\beta_1 = \sqrt{\alpha_1}, \quad \beta_2 = \sqrt{\alpha_2},$$
$$K = k(\beta_1, \beta_2).$$

On cherche à déterminer le degré de $[K : k]$. MAPLE met 10 secondes pour trouver que $X^2 - \alpha_1$ est irréductible sur $k$. On doit maintenant essayer de factoriser $X^2 - \alpha_2$ sur $k(\beta_1)$. Après 880 secondes, MAPLE retourne deux facteurs linéaires. On en déduit que $[K : k] = 2$.

```
# definition de k
> k:={RootOf(_Z^2-2), RootOf(_Z^2-3), RootOf(_Z^2+5)}:
> alpha1:=1+RootOf(_Z^2+5):
> alpha2:=1-RootOf(_Z^2+5):
# factorisation de X^2-alpha1
> evala(Factor(X^2-alpha1, k));
                              2                 2
                             X  - 1 - RootOf(_Z  + 5)
# kbeta1:=(beta1)
> kbeta1:=k union {RootOf(_Z^2-alpha1)}:
# factorisation de X^2-alpha2
> evala(Factor(X^2-alpha2, kbeta1));
   (X - 1/6 %3 %4 %2 %1 + 1/6 %4 %2 %1) (X + 1/6 %3 %4 %2 %1 - 1/6 %4 %2 %1)

                                        2
%1:=                           RootOf(_Z  - 2)

                                        2
%2:=                           RootOf(_Z  - 3)

                                        2
%3:=                           RootOf(_Z  + 5)

                                     2
%4:=                         RootOf(_Z  - 1 - %3)
```

### 2.2 L'algorithme de Zippel

Cet algorithme suppose deux choses [3] : la première que les entiers $r_i$ sont tous égaux à $r$, et deuxièmement que le corps $k$ contient les racines $r$-ièmes de l'unité.
On pose

$$\Delta = \{\alpha_1^{e_1}\alpha_2^{e_2}\dots\alpha_n^{e_n}, 0 \le e_i < r\},$$
$$k^{*r} = \{a^r, a \in k^*\},$$

et

$$\Delta k^{*r} = \{a^r b, a \in k^*, b \in \Delta\}.$$

Alors, on a le résultat suivant :

**Théorème.** (Kummer; Artin et Tate, 1968) Le groupe de Galois $\mathrm{Gal}(K/k)$ est isomorphe à $\Delta k^{*r}/k^{*r}$, et par suite $[K : k] = (\Delta k^{*r} : k^{*r})$.

Nous renvoyons le lecteur à [3] pour l'algorithme de calcul de $\Delta k^{*r}/k^{*r}$. On ne sait pas à l'heure actuelle comment se passer de la deuxième hypothèse de Zippel.
Reprenons l'exemple précédent. On part de

$$\Delta_1 = \{1, \alpha_1, \alpha_2, \alpha_1\alpha_2\}.$$

MAPLE a besoin de 10 secondes pour montrer que $X^2 - \alpha_1$ et $X^2 - \alpha_2$ sont irréductibles sur $k$, c'est-à-dire que ni $\alpha_1$, ni $\alpha_2$ ne sont des carrés parfaits dans $k$. Par contre, $X^2 - \alpha_1\alpha_2 = X^2 - 6$ a deux facteurs linéaires sur $k$ (après 22 secondes de calcul) et donc $\alpha_1\alpha_2$ est un carré dans $k$. On traduit cela par les relations $\alpha_1\alpha_2 \simeq 1 \bmod k^{*2}$, $\alpha_2^2 \simeq 1$ et par suite, $\alpha_1 \simeq \alpha_2$. On en déduit que

$$\Delta k^{*2}/k^{*2} = \{1, \alpha_1\}.$$

Et par suite, $[K : k] = 2$.

### 2.3 Implantation en MAPLE

L'auteur a implanté l'algorithme de Zippel; la fonction de simplification s'appelle `radnormal`. Remarquons au passage que l'on est amené à faire des choix de stratégie dans l'élaboration des tours de corps (cas de plusieurs extensions imbriquées). Il semble qu'en pratique, il vaut mieux faire un "grand" nombre de factorisations dans un "petit" corps, qu'une seule factorisation dans un "grand" corps.
Si l'on revient à l'exemple cité dans l'introduction, MAPLE, via la procédure `radnormal`, met maintenant 3 secondes à vérifier que les expressions obtenues par les formules de Cardan sont bien des racines de `p`.

## Références

[1] A. Borodin, R. Fagin, J. E. Hopcroft, and M. Tompa. Decreasing the nesting depth of expressions involving square roots. *Journal of Symbolic Computation*, 1:169–188, 1985.

[2] B. F. Caviness and R. J. Fateman. Simplification of radical expressions. In *Proceedings of the 1976 ACM Symposium on Symbolic and Algebraic Computation*, pages 329–338, 1976.

[3] R. Zippel. Simplification of expressions involving radicals. *Journal of Symbolic Computation*, 1:189–210, 1985.

# 8

# Introduction aux fonctions holonomes en une variable

Philippe Flajolet
INRIA, Rocquencourt

[résumé par Michèle Soria]

Les fonctions holonomes ou solutions d'équations différentielles linéaires à coefficients rationnels jouent un rôle dans l'analyse d'algorithmes liés aux structures ordonnées, en analyse combinatoire, dans la théorie des fonctions spéciales, et en analyse asymptotique. Cet exposé a pour objet les propriétés algébriques de base de ces fonctions, selon Stanley, Lipshitz et Zeilberger.

## 1 Introduction

La classe des fonctions et suites holonomes a été étudiée systématiquement depuis les années 80 par Stanley [11], Lipshitz [9] et Zeilberger [13]. Les riches propriétés du monde holonome en font un domaine central de l'analyse combinatoire et de la théorie des fonctions spéciales : propriétés de clôture, décidabilité de l'égalité et propriétés asymptotiques des suites holonomes. Les fonctions holonomes jouent un rôle important dans l'analyse d'algorithmes de recherche et de tri [2].

Une série $f(z) = \sum f_n z^n \in C[[z]]$ est dite *holonome* ("*D-finite*" dans [11]) si et seulement si elle satisfait une équation différentielle linéaire à coefficients rationnels ($\in \mathbb{C}(z)$) :

$$C_0(z)D^r f(z) + C_1(z)D^{r-1} f(z) + \ldots + C_r(z) f(z) = 0,$$

où $D = \frac{d}{dz}$. On peut en fait supposer que tous les $C_i$ sont des polynômes, ou que $C_0(z) = 1$.
A cette définition sur les séries correspond, modulo les conditions initiales, une définition équivalente sur les suites. Une suite $(f_n)$ est dite *holonome* si et seulement si elle satisfait une récurrence linéaire à coefficients polynomiaux :

$$d_0(n) f_{n+s} + d_1(n) f_{n+s-1} + \ldots + d_s(n) f_n = 0.$$

Exemple 1. Les graphes 2-réguliers (tous sommets d'arité 2) vérifient la récurrence

$$g_n = (n-1)\ g_{n-1} + \frac{(n-1)(n-2)}{2}\ g_{n-3},$$

avec les conditions initiales $g_0 = 1, g_1 = g_2 = 0$. Cette récurrence se traduit sur la série génératrice $g(z) = \sum g_n \frac{z^n}{n!}$ par l'équation différentielle $g'(z)(1-z) - \frac{1}{2}z^2 g(z) = 0$. Plus généralement, il a été montré par I. Gessel [5] que la série génératrice des graphes k-réguliers est une fonction holonome.

Exemple 2. La longueur moyenne de cheminement dans les arbres-quad vérifie la récurrence

$$p_n = e_n + 4 \sum_{k=0}^{n-1} \pi_{n,k}\ p_k,$$

avec $e_n = n$ et $\pi_{n,k} = \frac{1}{n}(H_n - H_k)$, où les $H_n$ sont les nombres harmoniques.
Ce type de récurrence est caractéristique des schémas de type *diviser pour régner* probabilistes, et apparaît dans l'analyse de Quicksort et des arbres de recherche multidimensionnels [3].

Exemple 3. La suite $u_n$ utilisée par Apéry dans la démonstration de l'irrationnalité de $\zeta(3)$

$$u_n = \sum_{k=0}^{n} \binom{n}{k}^2 \binom{n+k}{k}^2,$$

est une suite holonome, et c'est en général le cas pour les sommes de produits de binomiaux.

## 2 Propriétés de clôture

La définition d'holonomie peut se traduire en termes d'espace vectoriel de dimension finie. Une série $f(z)$ est holonome si et seulement si l'ensemble infini $\{f, Df, D^2f, \ldots\}$ de ses dérivées engendre un espace vectoriel de dimension finie sur $\mathbb{C}(z)$. Et la suite $(f_n)$ est holonome si et seulement si l'espace vectoriel des suites engendré sur $\mathbb{C}(n)$ par $\{f, Ef, E^2f, \ldots\}$ est fini ($Ef_n = f_{n+1}$).
Chacune des propriétés de clôture de la classe des fonctions (suites) holonomes est fondée sur la finitude d'un certain espace vectoriel, fermé par dérivation. Soit $V$ un espace vectoriel, $(\mathbf{b}_1, \ldots, \mathbf{b}_r)$ un système de générateurs, et $\mathbf{w}_0, \ldots, \mathbf{w}_r$ un système de $r+1$ vecteurs donnés par leurs coordonnées dans la base : $\mathbf{w}_j = \sum_{k=1}^{r} x_{j,k}\mathbf{b}_k$. La dépendance des $\mathbf{w}_i$ se traduit par l'équation

$$\begin{vmatrix} \mathbf{w}_0 & x_{0,1} & \ldots & x_{0,r} \\ \mathbf{w}_1 & x_{1,1} & \ldots & x_{1,r} \\ \ldots & \ldots & \ldots & \ldots \\ \mathbf{w}_r & x_{r,1} & \ldots & x_{r,r} \end{vmatrix} = 0.$$

Ce déterminant est une forme linéaire en les vecteurs $\mathbf{w}_0, \ldots, \mathbf{w}_r$. Et dans des espaces vectoriels engendrés par des fonctions et leurs dérivées, cette dépendance se traduit par une équation différentielle.

**Théorème de clôture** [1, 8, 9, 11, 13]
*(i) Toute fonction algébrique est holonome,*
*(ii) La classe des fonctions holonomes est fermée par les opérations de somme, produit de Cauchy et produit d'Hadamard,*
*(iii) Si $f$ est holonome et $g$ est algébrique, alors $f \circ g$ est holonome,*
*(iv) La classe des fonctions holonomes est fermée par dérivation, intégration, transformée de Laplace directe et inverse.*

Les fonctions $\exp(x)$, $\log(1+x)$, $\sin(x)$, $\cos(x)$, $\arcsin(x)$, $\arctan(x)$ sont holonomes ; mais ni $\tan(x)$, ni $1/\cos(x)$ ne sont holonomes : les fonctions holonomes ne sont fermées ni par composition ni par inverse.

Exemple 4. Fonctions algébriques : lorsque $y$ satisfait une équation algébrique de degré $d$, $y$ et toutes ses dérivées appartiennent à l'espace vectoriel engendré sur $\mathbb{C}(z)$ par $(1, y, \ldots, y^{d-1})$, et $y$ vérifie une équation différentielle homogène d'ordre au plus $d$. Par exemple la série génératrice des nombres de Motzkin vérifie l'équation algébrique $R(z, y(z)) = 0$, avec $R(z, y) = y - z(1 + y + y^2)$. En dérivant, on exprime $y'$ comme une fonction rationnelle en $z$ et $y$ : $y = N(y, z)/D(y, z)$. Pour réduire le dénominateur, on applique l'algorithme

de Bezout-Euclide aux polynômes $R$ et $D$, ce qui donne deux polynômes en $y$ à coefficients dans $\mathbb{C}(z)$, $U$ et $V$, tels que $UD - VR = 1$. D'où $y' \equiv N.U \mod R$. Pour les nombres de Motzkin, on obtient ainsi $y' = \big(-2z-(z-1)y\big)/\big(3z^3+2z^2-z\big)$, et donc $y$ vérifie une équation différentielle homogène d'ordre 2.

Exemple 5. Sommes et produits. Les séries $f(z) = \exp(z)$ et $g(z) = \log(1+z)$ sont holonomes : $f' - f = 0$ et $(1+z)g'' + g' = 0$. L'espace vectoriel des dérivées de la somme $h = f+g$ est engendré par la base $\{f, g, g'\}$. Le calcul de $h, h', h'', h'''$ dans cette base, et l'équation de déterminant qui résulte de la dépendance linéaire de $h, h', h''$ et $h'''$, donnent l'équation différentielle $(2+z)(1+z)h'''(z) - (-1+2z+z^2)h''(z) - (3+z)h'(z) = 0$. Pour la fonction produit $k = fg$, on utilise la base $\{fg, fg'\}$, et le calcul de $k, k', k''$ dans cette base donne $(1+z)k''(z) - (1+2z)k'(z) + zk(z) = 0$.

**Différences finies et sommes combinatoires.** La clôture par substitution algébrique entraîne l'holonomie d'une grande classe de suites et de fonctions : si

$$g_n = \sum_{k=0}^{n} \binom{n}{k} (-1)^k f_k \quad \Longleftrightarrow \quad g(z) = \frac{1}{1-z}\ f(-\frac{z}{1-z}),$$

alors $f_n$ est holonome lorsque $g_n$ est holonome. Par exemple la somme $\sum_{k=0}^{n} \binom{n}{k} \frac{(-1)^k}{1+k^2}$ est holonome d'ordre 3, car la suite $1/(1+k^2)$ est holonome d'ordre 3 (les substitutions rationnelles préservent l'ordre).

**Fonctions hypergéométriques.** La série hypergéométrique

$$F[a,b;c;z] = \sum_{n=0}^{\infty} \frac{\binom{-a}{n}\binom{-b}{n}}{\binom{-c}{n}}\ (-1)^n\ \frac{z^n}{n!}$$

vérifie l'équation différentielle $z(1-z)F''(z) + (c-(a+b+1)z)F'(z) - abF(z) = 0$.
De nombreuses identités entre coefficients binomiaux sont des traductions d'identités entre fonctions hypergéométriques. Par exemple la deuxième identité d'Euler

$$F[a,b;c;z] = (1-z)^{-a}\ F[a,c-b;c;-\frac{z}{1-z}]$$

se traduit par

$$(-1)^n\ \frac{\binom{-a}{n}\binom{-b}{n}}{\binom{-c}{n}}\ =\ \sum_{k=0}^{n} \frac{\binom{-a}{k}\binom{b-c}{k}\binom{n+a-1}{k+a-1}}{\binom{-c}{k}}.$$

Les identités sur les fonctions sont décidables d'après les propriétés de clôture. Les identités correspondantes sur les coefficients sont donc finiment décidables [13].

# 3 Propriétés asymptotiques

L'asymptotique des fonctions holonomes en une variable est un problème pour l'essentiel décidable, modulo les problèmes de périodicité. Les solutions d'équations différentielles à coefficients analytiques ont des singularités qui proviennent uniquement des coefficients de l'équation. Ces singularités sont classées en deux types : régulières et irrégulières. Dans le premier cas, les solutions ont un développement local singulier de la forme

$$(z-\rho)^{\alpha}\ \log^k(z-\rho).$$

Et dans le second cas le développement est du type

$$\exp\left(Q(\frac{1}{(z-\rho)^{p/q}})\right)\ (z-\rho)^{\alpha}\ \log^k(z-\rho),$$

où $Q$ est un polynôme.
Les méthodes d'analyse complexe, analyse de singularité [4] dans le premier cas et méthode de col [10] dans le second cas, permettent de traduire ces développements locaux en une information sur l'asymptotique des coefficients des fonctions solutions. D'où le résultat

**Théorème** [12] *Toute suite holonome $(f_n)$ est asymptotiquement équivalente à une somme de termes de la forme*

$$\lambda(n!)^{r/s}e^{Q(n^{1/m})}\omega^n n^{\alpha}(\log n)^k,$$

*où $r, s, m, k$ sont des entiers, $Q$ est un polynôme, et $\lambda, \omega, \alpha$ sont des nombres complexes.*

Exemple 6. La série génératrice du coût moyen d'une recherche partiellement spécifiée dans les arbres-quad [3] vérifie l'équation différentielle $(1-z)^2\ (zQ(z))'' - 4Q(z) = 2/(1-z)$. La fonction $Q(z)$ a une singularité régulière en $\rho = 1$, et un développement local équivalent à $(1-z)^{-\alpha}$, avec $\alpha = (\sqrt{17}-1)/2$. On en déduit par analyse de singularité que le coût moyen d'une recherche vaut asymptotiquement $Kn^{\alpha-1}$.

Exemple 7. Le nombre moyen de sous-suites croissantes dans les permutations [7] a une série génératrice exponentielle qui vérifie l'équation $(1-z)^2S'(z) + (z-2)S(z) = 0$. La fonction $S(z) = \frac{1}{1-z}e^{\frac{z}{1-z}}$ a une singularité irrégulière en $\rho = 1$, et l'on obtient par méthode de col que le nombre moyen de sous-suites croissantes dans une permutation de $[1 \ldots n]$ est équivalent à $\frac{1}{2\sqrt{e\pi}}n^{-1/4}e^{2\sqrt{n}}$.

# 4 Extensions

La notion d'holonomie s'étend aux fonctions à plusieurs variables : une fonction $f(z_1, \ldots, z_r)$ dans $C[[z_1, z_2, \ldots, z_r]]$ est dite *holonome* si l'ensemble de ses dérivées partielles

$$\frac{\partial^{j_1}}{\partial z^{j_1}}\frac{\partial^{j_2}}{\partial z^{j_2}}\cdots\frac{\partial^{j_r}}{\partial z^{j_r}}f(z_1, z_2, \ldots, z_r)$$

engendre un espace vectoriel de dimension finie sur le corps $\mathbb{C}(z_1, z_2, \ldots, z_r)$ des fractions rationnelles. De nombreuses propriétés de clôture sont conservées (voir le résumé de l'exposé de K. Compton page 47 et suivantes).

La théorie des fonctions holonomes permet de décider (et de découvrir) des identités sur les fonctions spéciales. Par exemple l'identité de Ramanujan

$$\frac{1}{4!}\ (\arcsin x)^4\ \equiv\ \sum_{k=2}^{\infty}\ \frac{4^{k-1}}{\binom{2k}{k}}\left(\frac{1}{1^2}+\frac{1}{2^2}+\ldots+\frac{1}{(k-1)^2}\right)\ \frac{x^{2k}}{(2k)^2}.$$

Les membres gauche et droit sont des fonctions appartenant à un espace vectoriel de dimension finie (ici 24), et il suffit de vérifier l'identité pour un nombre fini de conditions initiales.
Une telle vérification peut être faite de façon automatique : des algorithmes ont été donnés par Zeilberger [13] et Gosper [6] dans le cas des sommes hypergéométriques.

Du point de vue de l'asymptotique, la théorie des fonctions holonomes fournit une équation différentielle de série génératrice sous une forme normalisée. L'analyse asymptotique se développe alors à partir d'une analyse de singularité (cas d'une singularité régulière) ou d'une analyse de col (cas d'une singularité irrégulière).

## Références

[1] L. Comtet. Calcul pratique des coefficients de Taylor d'une fonction algébrique. *Enseignement Math.*, 10:267–270, 1964.

[2] Ph. Flajolet. Analytic analysis of algorithms. In W. Kuich, editor, *Automata, Languages and Programming*, number 623 in Lecture Notes in Computer Science, pages 186–210, 1992. Proceedings of the 19th International Colloquium, Vienna, July 1992.

[3] Ph. Flajolet, G. Gonnet, C. Puech, and J. M. Robson. The analysis of multidimensional searching in quad–trees. In *Proceedings of the Second Annual ACM–SIAM Symposium on Discrete Algorithms*, pages 100–109, Philadelphia, 1991. SIAM Press.

[4] Ph. Flajolet and A. M. Odlyzko. Singularity analysis of generating functions. *SIAM Journal on Discrete Mathematics*, 3(2):216–240, 1990.

[5] I. M. Gessel. Symmetric functions and $P$–recursiveness. *Journal of Combinatorial Theory,* Series A, 53:257–285, 1990.

[6] R. W. Gosper. Decision procedure for indefinite hypergeometric summation. *Proceedings of the National Academy of Sciences USA*, 75(1):40–42, January 1978.

[7] V. Lifschitz and B. Pittel. The number of increasing subsequences of the random permutation. *Journal of Combinatorial Theory,* Series A, 31:1–20, 1981.

[8] L. Lipshitz. The diagonal of a $D$-finite power series is $D$-finite. *J. Algebra*, 113:373–378, 1988.

[9] L. Lipshitz. $D$-finite power series. *J. Algebra*, 122:353–373, 1989.

[10] L. Sirovich. *Techniques of Asymptotic Analysis*. Springer Verlag, 1971.

[11] R. P. Stanley. Differentiably finite power series. *European Journal of Combinatorics*, 1:175–188, 1980.

[12] J. Wimp and D. Zeilberger. Resurrecting the asymptotics of linear recurrences. *J. Math. Anal. Appl.*, 111:162–176, 1985.

[13] D. Zeilberger. A holonomic approach to special functions identities. *Journal of Computational and Applied Mathematics*, 32:321–368, 1990.

# 9

# Fonctions holonomes à plusieurs variables

Kevin Compton
University of Michigan and INRIA, Rocquencourt

[résumé par Philippe Robert]

Résumé. Cet exposé est la suite de celui sur les fonctions holonomes à une variable (page 41). Une série à plusieurs variables est dite holonome (ou D-finie) lorsque ses dérivées partielles engendrent un espace vectoriel de dimension finie sur le corps des fonctions rationnelles complexes. La plupart des fonctions spéciales rencontrées en analyse d'algorithmes sont holonomes. Les travaux de L. Lipshitz et D. Zeilberger sur les propriétés de clôture des fonctions holonomes sont exposés, ainsi qu'un algorithme dû à Zeilberger pour décider d'identités entre séries. Cet algorithme est basé sur une forme normale des fonctions holonomes.

## 1 Les suites et les fonctions holonomes à une variable

Dans le cas d'une variable (cf. l'exposé de Ph. Flajolet sur les fonctions holonomes), la correspondance suite-fonction holonome est résumée par le tableau suivant :

| | **Suites** | **Fonctions** |
|---|---|---|
| Représentation | $f = (f_n)_{n\in\mathbb{N}}$ | $f(z) = \sum_{n\in\mathbb{N}} f_n z^n$ |
| Opérations | $E(f) = (f_{n+1})_n$ | $Z(f)(z) = \sum f_n z^{n+1}$ |
| | $N(f) = (nf_n)_n$ | $D(f)(z) = \sum nf_n z^n$ |
| Propriétés | P-récursivité | D-finitude |
| | $P(E, N)(f) = 0$ | $Q(Z, D)(f) = 0$ |

$P, Q$ polynômes sur $\mathbb{C}$

De façon équivalente, une fonction $f(z)$ est D-finie si et seulement si le sous espace vectoriel sur le corps des fractions rationnelles $\mathbb{C}(Z)$ engendré par les $D^n(f), n \in \mathbb{N}$ est de dimension finie. Cette propriété-définition donne la clé de la plupart des propriétés de clôture de l'ensemble des fonctions holonomes à une variable. Dans le cas des fonctions à plusieurs variables, elle permet en outre une extension naturelle de la définition de D-finitude.

## 2 Séries génératrices holonomes à plusieurs variables

**Définition 1** *Une fonction $f(z_1, \ldots, z_n)$ est holonome (ou D-finie) si le sous espace vectoriel sur $\mathbb{C}(Z)$ engendré par les $D^\alpha(f)$, $\alpha \in \mathbb{N}^k$ est de dimension finie, avec*

$$D^\alpha(f) = D_1^{\alpha_1} \ldots D_k^{\alpha_k} f(z_1, \ldots, z_k), \qquad \alpha = (\alpha_1, \ldots, \alpha_n), D_i = \frac{\partial}{\partial z_i}.$$

Si $f = (f_\alpha)_\alpha$ est une suite de $\mathbb{C}^k$ et $f(z) = \sum_\alpha f_\alpha z^\alpha$ sa fonction génératrice (avec $z^\alpha = \prod z_i^{\alpha_i}$), alors $f$ est D-finie si et seulement si $f$ vérifie un système d'équations différentielles non triviales,

$$P_i(Z_1, \ldots, Z_k, D_i)(f) = 0, i = 1, \ldots, k.$$

les $P_i$ étant des polynômes et $Z_i$ l'opérateur défini par $Z_i(f)(z) = z_i f(z)$.
La plupart des propriétés de clôture valables en dimension 1 s'étendent de la même manière au cas à plusieurs variables :

**Proposition 1**

*(i) Les fonctions algébriques sont D-finies.*

*(ii) Si $f, g$ sont D-finies, il en va de même pour $f + g$ et $fg$.*

*(iii) Si*

$$f(z) = \sum_{\alpha \in \mathbb{N}^j,\ \beta,\gamma \in \mathbb{N}^k} f_{\alpha\beta\gamma} z^{\alpha\beta\gamma}$$

*est D-finie (avec $\alpha\beta\gamma$ la concaténation de $\alpha, \beta, \gamma$), alors*

$$g(z) = \sum_{\alpha \in \mathbb{N}^j,\ \beta \in \mathbb{N}^k} f_{\alpha\beta\beta} z^{\alpha\beta\beta}$$

*l'est aussi ($g$ est appelée une diagonale de $f$).*

*(iv) Si $f(z_1, \ldots, z_k)$ est D-finie et les fonctions $g_1, \ldots, g_k$ algébriques alors $f(g_1, \ldots, g_k)$ est D-finie.*

L'assertion *(iii)* du théorème précédent entraîne en particulier la propriété de clôture par produit de Hadamard des fonctions holonomes :

$$\left(\sum_n f_n z_1^n\right)\left(\sum_n g_n z_2^n\right) = \sum_{mn} f_m g_n z_1^m z_2^n = F(z_1, z_2),$$

il suffit de remarquer que le produit de Hadamard de $f$ et $g$, $\sum_n f_n g_n z^n$ est une diagonale de $F$. Lipshitz (1988) a montré de cette façon que le produit de Hadamard est D-fini quand chacune des deux fonctions l'est, sa procédure algorithmique pour déterminer le polynôme annulateur est toutefois assez lourde. Une autre solution consiste à suivre la procédure de Stanley pour le cas à une dimension : utiliser la partie discrète (avec les opérateurs $E$ et $N$) de la définition d'holonomie pour montrer la clôture par produit d'Hadamard.

## 3 Suites holonomes à plusieurs variables

**Définition 2** *Une $m$-section d'une suite $(f_\alpha)_{\alpha \in \mathbb{N}^k}$ est une sous-suite de $f$, $(f_\alpha)_{\alpha \in \Delta}$ avec $\Delta = \{\alpha \in \mathbb{N}^k / \alpha_{i_1} = n_1, \ldots, \alpha_{i_p} = n_p\}$ et $i_1, \ldots, i_p \leq m$.*

**Définition 3** *Une suite $(f_\alpha)_{\alpha \in \mathbb{N}^k}$ est dite holonome (ou P-récursive) si il existe un entier $m$ et un système d'équations polynomiales non triviales,*

$$P_i(N_i, E_1, \ldots, E_k)(f) = 0, \quad i = 1, \ldots, k$$

*vérifiant*

*(i) le degré en $E_j$ de $P_i$ est $\leq m$.*

*(ii) Toutes les m-sections de $f$ sont P-récursives.*

La correspondance entre suites et fonctions est assurée par la proposition suivante due à Lipshitz,

**Proposition 2** *La suite $(f_\alpha)_{\alpha\in\mathbb{N}^k}$ est P-récursive si et seulement si $\sum f_\alpha z^\alpha$ est D-finie.*

## References

[1] L. Lipshitz. The diagonal of a $D$-finite power series is $D$-finite. *J. Algebra*, 113:373–378, 1988.

[2] L. Lipshitz. $D$-finite power series. *J. Algebra*, 122:353–373, 1989.

[3] R. P. Stanley. Differentiably finite power series. *European Journal of Combinatorics*, 1:175–188, 1980.

[4] D. Zeilberger. A holonomic approach to special functions identities. *Journal of Computational and Applied Mathematics*, 32:321–368, 1990.

# 10

# Holonomic Symmetric Functions

Dominique Gouyou-Beauchamps
LRI, Orsay

[summary by Philippe Flajolet]

By suitably defining a notion of holonomy for symmetric functions in infinitely many variables, Gessel [3] derives or rederives the holonomic character of generating functions for structures as diverse as: *$k$–regular graphs; Young tableaux of fixed height $k$; permutations with longest increasing subsequence of length $k$; square integer matrices with row sum and column sum $k$; $k \times n$ latin rectangles.*
Each of these problems represents a highly non trivial enumeration problem that is far from being amenable to direct symbolic methods (except for simple boundary cases).
Recall that a univariate function is *holonomic* if it satisfies a linear differential equation with polynomial coefficients (this is also called $D$–finite). Accordingly, its coefficients, in turn named holonomic, satisfy recurrences with polynomial coefficients (this is also called $P$–recursive).

**Symmetric functions.** We work with infinitely many indeterminates $x_1, x_2, \ldots$, and we are concerned with particular *symmetric functions*.
(a). If $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_k)$ is a partition, then define the *monomial* symmetric function

$$m_\lambda = \sum_{i_1, i_2, \ldots, i_k} x_{i_1}^{\lambda_1} x_{i_2}^{\lambda_2} \cdots x_{i_k}^{\lambda_k}, \tag{1}$$

the sum being extended to all distinct indices. For instance, if $\lambda = (3,1,1) \equiv (3,1^2)$, then $m_\lambda = \sum x^3 yz$.
(b). The *elementary* symmetric functions are

$$e_r = \sum_{i_1 < i_2 < \cdots < i_r} x_{i_1} x_{i_2} \cdots x_{i_r}, \tag{2}$$

so that

$$E(t) := \sum_{r \geq 0} e_r t^r = \prod_{i \geq 1} (1 + x_i t). \tag{3}$$

This is extended to partitions indices: $e_\lambda = e_{\lambda_1} e_{\lambda_2} \cdots e_{\lambda_k}$. For instance, $e_{(3,3,1,1,1)} = e_3^2 e_1^3$.
(c). The *complete* symmetric functions are

$$h_r = \sum_{i_1 \leq i_2 \leq \cdots \leq i_r} x_{i_1} x_{i_2} \cdots x_{i_r}, \tag{4}$$

so that

$$H(t) := \sum_{r \geq 0} h_r t^r = \prod_{i \geq 1} (1 - x_i t)^{-1}. \tag{5}$$

One similarly defines $h_\lambda = h_{\lambda_1} h_{\lambda_2} \cdots h_{\lambda_k}$.
(d). The *power* symmetric functions are

$$p_r = \sum_i x_i^r \tag{6}$$

so that

$$P(t) := \sum_r p_r t^r = \sum_i \frac{t x_i}{1 - t x_i}. \tag{7}$$

Again the definition extends to $p_\lambda = p_{\lambda_1} p_{\lambda_2} \cdots p_{\lambda_k}$.
Each of the sets $\{e_\lambda\}$, $\{h_\lambda\}$, $\{m_\lambda\}$, $\{p_\lambda\}$ constitutes a basis for symmetric functions. Formulæ for changing bases derive from the generating functions (3,5,7).

**Operations.** (a). One classically defines a *scalar product* with orthogonality properties

$$(P_1): \quad \langle m_\lambda, h_\mu \rangle = \delta_{\lambda,\mu}; \qquad (P_2): \quad \langle p_\lambda, p_\mu \rangle = z_\lambda \delta_{\lambda,\mu}, \tag{8}$$

where $z_\lambda = 1^{r_1} 2^{r_2} \cdots k^{r_k} r_1! r_2! \cdots r_k!$ if $\lambda = (1^{r_1} 2^{r_2} \cdots k^{r_k})$.
(b). The *internal* (or *inner*) product is defined by

$$p_\lambda * p_\mu = \delta_{\lambda,\mu} z_\lambda p_\lambda. \tag{9}$$

It is like the scalar product except that it keeps track of variables.
(c). Finally, there is an operation of *composition* (also known as *plethysm*) written $f(g)$ and defined by $p_m(p_n) = p_{mn}$.

**The example of regular graphs.** The scalar product permits us to extract coefficients: If $f$ is symmetric, then the coefficient of $x_{i_1}^{\lambda_1} \cdots x_{i_k}^{\lambda_k}$ in $f$ is up to normalization the coefficient of $m_\lambda$ in $f$ which equals [by $(P_1)$ of Eq. (8)] $\langle f, h_\lambda \rangle$. Given that $f$ is symmetric, it can be expressed in terms of the power functions $p_r$, as well as $h_\lambda$. The computation of the coefficient then reduces to the computation of a scalar product which [by $(P_2)$ of Eq. (8)] is a modified Hadamard product of two functions.
Let us work out the example of 2–*regular graphs*. The function

$$G(x;t) = \prod_{i<j} (1 + x_i x_j t) \tag{10}$$

is a generating function in *infinitely many variables* that encodes all the graphs over a denumerable collection of vertices. (Vertices are labelled and vertex $i$ is represented by the indeterminate $x_i$. Thus $x_i x_j$ encodes the edge connecting $i$ and $j$.) The additional variable $t$ records the number of edges. It is a simple exercise to express $G$ in terms of the $p_n = \sum_i x_i^n$:

$$\log G = \sum_{n=1}^{\infty} (-1)^{n-1} \frac{t^n}{n} \left( \sum_{i<j} x_i^n x_j^n \right),$$

so that

$$G(x;t) = \exp\left( \frac{1}{2} \sum_{n=1}^{\infty} \frac{t^n}{n} (p_n^2 - p_{2n}) \right). \tag{11}$$

The number $R_n$ of labelled 2–regular graphs on $n$ nodes is given by

$$R_n t^n = [x_1^2 x_2^2 \cdots x_n^2] G(x;t).$$

From the definition of $m_\lambda$, we have $[x_1^2 \cdots x_n^2] m_{(2^n)} = n!$, thus

$$R_n t^n = n! \cdot [\text{coeff. of } m_{(2^n)}]\, G(x;t),$$

and by orthogonality [see $(P_1)$]:

$$R_n \frac{t^n}{n!} = \langle G(x;t), h_{(2^n)} \rangle = \langle G(x;t), h_2^n \rangle. \tag{12}$$

Quantity $h_2$ is itself expressible in the $p$'s: $h_2 = \frac{1}{2}p_1^2 + \frac{1}{2}p_2$. Thus:

$$R_n \frac{t^n}{n!} = \langle G(x;t), (\frac{1}{2}p_1^2 + \frac{1}{2}p_2)^n \rangle. \tag{13}$$

In this scalar product, only the part of $G$ that involves $p_1$ and $p_2$ counts. In other words,

$$R_n \frac{t^n}{n!} = \langle \exp(\frac{t}{2}(p_1^2 - p_2) + \frac{t^2}{4}p_2^2), (\frac{1}{2}p_1^2 + \frac{1}{2}p_2)^n \rangle. \tag{14}$$

This already provides a combinatorial expression (a sum of finite "rank" in the sense of Comtet [2, p. 216]) for $R_n$ upon expanding and using $(P_2)$. The inner product can also be formulated as a Hadamard product with respect to $\{p_1, p_2\}$ and $t$. One finally needs to generate the coefficients $z_\lambda$ with is achieved by means of a further Hadamard product with

$$\Phi(p_1)\Phi(2p_2) \qquad \text{where} \qquad \Phi(z) = \sum_{n=0}^{\infty} n! z^n.$$

Summarizing, we have obtained[3]

$$\sum_n R_n \frac{t^n}{n!} = \left[ \exp(\frac{t}{2}(p_1^2 - p_2) + \frac{t^2}{4}p_2^2) \odot_{t,p_1,p_2} \frac{1}{1 - t(\frac{1}{2}p_1^2 + \frac{1}{2}p_2)} \odot_{p_1,p_2} \Phi(p_1)\Phi(2p_2) \right]_{p_1,p_2 \mapsto 1}. \tag{15}$$

The process is of course fully general.

**Theorem 1** *For any fixed $k$, the exponential generating function of $k$–regular graphs is holonomic and expressible as a Hadamard product*

$$\exp(E(t;\mathbf{p})) \odot_{t,\mathbf{p}} R(\mathbf{p}) \odot_{\mathbf{p}} \prod_{j=1}^{k} \Phi(jp_j) \Big|_{\mathbf{p} \mapsto \mathbf{1}},$$

*for some effectively computable polynomial $E$ and rational function $R$.*

Notice that the number of $r$–regular labelled graphs on $n$ vertices satisfies (see Bollobás's book [1, II.4]):

$$R_n^{(k)} \sim \sqrt{2} e^{-(k^2-1)/4} \left( \frac{k^{k/2}}{e^{k/2} k!} \right)^n n^{kn/2}.$$

[3] The enumeration of 2–regular graphs is accessible to symbolic methods and it is easily found directly (see [2, p. 272]) that the exponential generating function is

$$\frac{e^{-t/2 - t^2/4}}{\sqrt{1-t}}.$$

**Coefficient extraction.** The example of regular graphs points to a general methodology by which coefficients of sufficiently "regular" monomials inside huge symmetric functions are extracted using a combination of change of bases and inner or scalar products. This extends techniques used earlier by Read, Goulden and Jackson and others. A typical statement is:

**Theorem 2** *The coefficient* $[x_1 x_2 \cdots x_n]$ *in the symmetric function* $f$ *is computable by its generating function,*

$$\sum_{n=0}^{\infty} \{[x_1 x_2 \cdots x_n]\, f\} \frac{X^n}{n!} = f(X, 0, 0, 0, \ldots),$$

*where* $f(p_1, p_2, \ldots)$ *is the expression of* $f$ *in terms of the* $p$*'s.*

This is Theorem 1 of [3]. A similar theorem gives the bivariate generating function for the coefficient of $x_1 \cdots x_m x_{m+1}^2 \cdots x_{m+n}^2$ in $f$ (Theorem 2 of [3]) or $[x_1 \cdots x_m x_{m+1}^3 \cdots x_{m+n}^3]\, f$ (Theorem 3 of [3]), and so on.
There are applications to the area of order patterns (up and down patterns, increasing subsequences) in permutations with repetitions of various types. For instance, the symmetric generating function of up–and–down sequences is seen to be (use inclusion–exclusion)

$$E = 1/\sum_{r=0}^{\infty} (-1)^r h_{2r},$$

from which counting results derive for alternating permutations of multisets that involve the secant numbers.

**Holonomy.** A function in the finite set of variables $x$ is holonomic if and only if the vector space spanned over $\mathbb{C}(x)$ by its derivatives is finite dimensional. We shall say that a symmetric function $f$ is holonomic (or $D$–finite) if and only if, as a function of the *infinite* set $p$'s, it is holonomic in any finite combination of the $p$'s. Most standard holonomy closure properties carry over to this definition provided infinite operations are carefully avoided.
Holonomic symmetric functions appear to be closed under inner product, scalar product with mild restrictions, and certain forms of plethysm. The proofs present no difficulty but have to relie on the closure of standard holonomic functions under Hadamard products (or diagonals), which is Lipshitz's "hard" theorem.
This concept constitutes the natural abstract setting in which results of the previous sections can be cast.

**Extensions.** Gessel's paper is very rich. It contains a discussion, based on Schur functions, of Young tableaux of bounded height. (The Schur function $s_\lambda$ is defined as

$$s_\lambda = \det(h_{\lambda_i - i + j})_{1 \le i,j \le n},$$

where we take $h_n = 0$ for $n < 0$.) It continues with symmetric functions in two sets (or more!) of variables, with applications to longest increasing subsequences in permutations.
The following theorem is notable:

**Theorem 3** *(i). The exponential generating function of the Young tableaux of height at most $k$ is expressible in terms of a determinant of Bessel functions.*
*(ii). The doubly exponential generating function for permutations with longest increasing subsequence of length at most $k$ is expressible in terms of a determinant of Bessel functions.*

Note: It was proved by Regev using the hook formula of tableaux and the Selberg integral that the number of permutations of $n$ with longest increasing subsequence of length at most $k$ satisfies asymptotically

$$L_n^{(k)} \sim \gamma_k k^{2n}, \qquad \gamma_k \in \mathbb{R}.$$

**Direct asymptotics.** Direct asymptotic approximations may well result from symmetric functions, after it is realized how different variables induce weights of different orders of growth. An example suggested by Gessel is the result of Everett and Stein: the number of $n \times n$ non–negative matrices with every row and column sum $k$ is $\langle h_k^n, h_k^n \rangle$, and it satisfies asymptotically

$$M_n^{(k)} \sim \frac{(kn)!}{(k)!^{2n}} e^{(k-1)^2/2}.$$

## References

[1] B. Bollobás. *Random Graphs.* Academic Press, 1985.

[2] L. Comtet. *Advanced Combinatorics.* Reidel, Dordrecht, 1974.

[3] I. M. Gessel. Symmetric functions and $P$–recursiveness. *Journal of Combinatorial Theory,* Series A, 53:257–285, 1990.

# 11

# L'algorithme de Kovačić

Michèle Loday-Richaud
Université de Paris-Sud, Orsay

[résumé par Philippe Le Chenadec]

## 1 Extensions liouvilliennes

### 1.1 Résumé de la théorie de Galois différentielle

Une extension de Picard-Vessiot du corps des fractions rationnelles $\mathbb{C}(x)$ est obtenue en lui adjoignant les solutions d'une équation différentielle ordinaire, linéaire et homogène (EDOLH), dont les coefficients appartiennent à $\mathbb{C}(x)$. Un élément d'un tel corps est liouvillien s'il est exprimable par combinaisons successives d'intégrales, d'exponentielles et d'éléments algébriques, les éléments de base appartenant à $\mathbb{C}(x)$. L'algorithme de Kovačić fournit les solutions liouvilliennes d'une EDOLH du second ordre lorsqu'il en existe. Des extensions à l'ordre quelconque ont été décrites par M. Singer dans [10]. Nous résumons ici la version originale de l'algorithme à l'ordre deux [6]. Pour des améliorations et de nombreux exemples, nous renvoyons aux travaux de A. Duval et M. Loday-Richaud dans [2].
La notion fondamentale est celle de corps différentiel : soit $K$ un corps, supposé de caractéristique $\mathrm{car}(K)$ nulle, une application $\partial : K \to K$ est une dérivation si et seulement si $\partial(x+y) = \partial(x)+\partial(y)$ et $\partial(xy) = \partial(x)y + x\partial(y)$, $x, y \in K$. Une dérivation annule le corps premier, ici $\mathbb{Q}$, et l'ensemble $C$ des *constantes* $x \in K$, $\partial(x) = 0$, est un sous-corps de $K$. Ce dernier est alors muni d'une structure de $C$-espace vectoriel qui fait de $\partial$ une application $C$-linéaire. Pour les EDOLH qui nous intéressent, nous avons $C = \mathbb{C}$ et $\partial = \frac{\mathrm{d}}{\mathrm{dx}}$, la dérivation usuelle.
Étant donnés deux corps différentiels $K$ et $L$, un morphisme de corps $\sigma : K \to L$ est différentiel s'il commute avec les dérivations : $\sigma \circ \partial_K = \partial_L \circ \sigma$. Une extension $K/k$ est dite différentielle lorsque $\partial_k = \partial_K | k$ (l'inclusion de $k$ dans $K$ est différentielle). On note $k\langle y_1, \ldots, y_n\rangle$ le sous-corps différentiel de $K$ engendré par $y_1, \ldots, y_n \in K$ (qui existe puisque l'intersection de sous-corps différentiels de $K$ est un sous-corps différentiel de $K$). Un opérateur différentiel $D$ défini sur $k$ est un polynôme non nul et non commutatif $D = a_n\partial^n + \cdots + a_0$ de $k[\partial]$. L'équation différentielle associée est l'équation $D(y) = a_n\partial^n(y) + \cdots + a_0 y = 0$, son degré est celui de $D$. Une solution de $D$ dans $K$ est un élément $x \in K$ tel que $D(x) = 0$. Dorénavant, nous supposerons $D$ unitaire.

**Définition 1** *L'extension différentielle $K/k$ est une extension de Picard-Vessiot associée à l'opérateur différentiel $D \in k[\partial]$, de degré $n$, si et seulement si $K$ et $k$ ont même corps $C$ de constantes, et $K = k\langle y_1, \ldots, y_n\rangle$, où les $y_i \in K$ sont des solutions de $D$ dans $K$, $C$-linéairement indépendantes.*

**Théorème 1** [5] *Si $\mathrm{car}(k) = 0$, $k$ corps différentiel à corps de constantes algébriquement clos, il existe une extension de Picard-Vessiot associée à tout opérateur différentiel à coefficients dans $k$, unique à $k$-isomorphisme différentiel près.*

On suppose désormais que le corps des constantes est algébriquement clos. Si $k \subset L \subset K$ sont des extensions différentielles, avec $K/k$ de Picard-Vessiot associée à $D$, alors $K/L$ est aussi de Picard-Vessiot, associée à $D$. Pour $K/k$ une extension différentielle, on définit le *groupe* de Galois différentiel $\mathrm{Gal}(K/k) = \{\sigma \in Aut(K) \mid \sigma \text{ est un } k-\text{automorphisme différentiel de } K\}$. Si de plus $K/k$ est l'*extension* de Picard-Vessiot associée à $D$, de degré $n$, tout morphisme $\sigma \in \mathrm{Gal}(K/k)$ donne par restriction un élément de $GL(V)$, où $V \subset K$ est le $C$-espace vectoriel de dimension $n$ des solutions dans $K$ de $D$, puisque $\sigma$ commute avec $\partial$. De plus, l'application $\mathrm{Gal}(K/k) \to GL(V)$ ainsi définie est un morphisme injectif, $K/k$ étant de Picard-Vessiot. Le choix d'une $C$-base de $V$ fournit donc une représentation linéaire $\mathrm{Gal}(K/k) \to GL_n(C)$, définie à conjugaison près dans $GL_n(C)$. Enfin, dernier ingrédient différentiel, l'extension $K/k$ est dite normale si et seulement si pour tout $x \in K - k$, il existe $\sigma \in \mathrm{Gal}(K/k)$ qui ne fixe pas $x$ : $\sigma(x) \neq x$. On démontre que toute extension de Picard-Vessiot à corps de constantes algébriquement clos est normale (esquisse de preuve dans l'article de M. Singer dans [10]). Rappelons qu'un sous-groupe de $GL_n(C)$ est ($C$-)algébrique s'il est fermé dans la topologie de Zariski de $GL_n(C)$, induite par la topologie de Zariski de l'espace affine $C^{n^2+1}$ via le plongement usuel du groupe linéaire dans cet espace.
Si l'extension $K/k$ est différentielle, pour tout sous-groupe $H \subset \mathrm{Gal}(K/k)$ on définit le sous-corps *différentiel* $K_H = \{x \in K \mid \forall \sigma \in H,\ \sigma(x) = x\}$, et à tout sous-corps différentiel $L \subset K$, on associe le sous-groupe $\mathrm{Gal}(K/L)$ de $\mathrm{Gal}(K/k)$.

**Théorème 2** *Soit $K/k$ une extension de Picard-Vessiot associée à l'opérateur différentiel $D$ de degré $n$, de corps de constantes $C$ algébriquement clos,* $\mathrm{car}(C) = 0$. *Le sous-groupe* $\mathrm{Gal}(K/k)$ *de $GL_n(C)$ est algébrique. Les correspondances définies ci-dessus sont des bijections naturelles réciproques entre l'ensemble des extensions différentielles intermédiaires $L$, $k \subset L \subset K$, et l'ensemble des sous-groupes algébriques $H \subset \mathrm{Gal}(K/k)$. L'extension $L/k$ est normale si et seulement si $H$ est normal dans* $\mathrm{Gal}(K/k)$, *auquel cas* $\mathrm{Gal}(L/k) = \mathrm{Gal}(K/k)/H$. *Pour tout sous-groupe $H$ de* $\mathrm{Gal}(K/k)$, *on a* $\mathrm{Gal}(K/K_H) = \overline{H}$ *(clôture de $H$ pour la topologie de Zariski).*

## 1.2 Extensions liouvilliennes et groupes résolubles

Rappelons tout d'abord quelques propriétés bien connues des groupes algébriques. Si $G$ est un tel groupe, la composante connexe $G^\circ$ de l'identité est un sous-groupe normal d'indice fini. Les groupes dérivés $D^0(G) = G$, $D^{i+1}(G) = [D^i(G), D^i(G)]$ sont algébriques, ce qui permet de définir les groupes algébriques résolubles exactement comme le sont les groupes résolubles abstraits : $D^i(G)$ est trivial pour $i$ assez grand. On a alors le théorème de Lie-Kolchin : tout groupe $C$-algébrique connexe résoluble $G$ est triangularisable, i.e. si $G \subset GL_n(C)$, il existe $\sigma \in GL_n(C)$ tel que $\sigma G \sigma^{-1} \subset T_n(C)$, le groupe algébrique des matrices triangulaires supérieures inversibles.
Le groupe algébrique $SL_n(C) = \{\sigma \in GL_n(C) | \mathit{dét}(\sigma) = 1\}$, $n > 1$, n'est pas résoluble (en fait, il est algébriquement simple : il ne possède pas de sous-groupe normal connexe propre). Avec la stabilité sous décomposition multiplicative de Jordan des groupes algébriques, ces résultats suffisent pour classifier les sous-groupes algébriques de $SL_2(\mathbb{C})$. On note $D_n(C)$ le groupe algébrique des matrices diagonales inversibles d'ordre $n$, de déterminant égal à 1, $S_n$ (resp. $A_n$) le groupe symétrique (resp. alterné) de degré $n$, et on pose $J = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$.

**Proposition 1** [4, 6] *Soit $G$ un sous-groupe algébrique de $SL_2(\mathbb{C})$, on a l'une des possibilités exclusives suivantes :*

1. *$G$ est triangularisable (dans $SL_2(\mathbb{C})$),*
2. *$G$ est conjugué à un sous-groupe de $D_2(\mathbb{C}) \cup J.D_2(\mathbb{C})$ et le cas 1 est exclu,*
3. *$G$ est fini et 1, 2 sont exclus,*
4. *$G = SL_2(\mathbb{C})$.*

Cette proposition fournit l'ossature de l'algorithme de Kovačić. Le cas 3 est complété par la détermination des sous-groupes finis de $SL_2(\mathbb{C})$, qui remonte pour l'essentiel à Fuchs, Jordan et Klein. La preuve donnée par Kovačić dans [6] est concise et élégante. On note $D^+$ le groupe $D_2(\mathbb{C}) \cup J.D_2(\mathbb{C})$, et $A(c) = \begin{pmatrix} c & 0 \\ 0 & c^{-1} \end{pmatrix}$.

**Proposition 2** *Un sous-groupe fini de $SL_2(\mathbb{C})$, non conjugué à un sous-groupe de $D^+$, est conjugué à l'un des groupes $G$ suivants, où $H$ désigne le sous-groupe de $G$ formé des matrices scalaires :*

1. *Cas tétraédrique, $G/H \simeq A_4$, $G$ est engendré par les matrices $A(\xi)$ et $\phi \begin{pmatrix} 1 & 1 \\ 2 & -1 \end{pmatrix}$, $\xi$ racine primitive sixième de l'unité, $3\phi = 2\xi - 1$.*
2. *Cas octaédrique, $G/H \simeq S_4$, $G$ est engendré par les matrices $A(\xi)$ et $\phi \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$, $\xi$ racine primitive huitième de l'unité, $2\phi = \xi(\xi^2 - 1)$.*
3. *Cas icosaédrique, $G/H \simeq A_5$, $G$ est engendré par les matrices $A(\xi)$, $\begin{pmatrix} \phi & \psi \\ \psi & -\phi \end{pmatrix}$, $\xi$ racine primitive dixième de l'unité, $5\phi = 3\xi^3 - \xi^2 + 4\xi - 2$, $5\psi = \xi^3 + 3\xi^2 - 2\xi + 1$.*

Soit $K/k$ une extension différentielle. Un élément $x \in K$ est dit primitif sur $k$ si $\partial(x) \in k$, il est dit exponentiel sur $k$ si $x^{-1}\partial(x) \in k$. Une extension $K/k$ est liouvillienne lorsque $K = k\langle x_1, \ldots, x_n \rangle$, avec $x_{i+1}$ algébrique, primitif ou exponentiel sur $k\langle x_1, \ldots, x_i \rangle$, $i = 1, \ldots, n-1$.

**Proposition 3** [5] *Soit $K/k$ une extension de Picard-Vessiot,* $\mathrm{car}(k) = 0$*, et $C$ algébriquement clos. Le corps $K$ est liouvillien sur $k$ si et seulement si la composante connexe de l'identité de* $\mathrm{Gal}(K/k)$ *est résoluble.*

Considérons une EDOLH à coefficients dans $\mathbb{C}(x)$ :

$$y'' + ay' + by = 0. \qquad (E)$$

Si $\eta$ est une solution non nulle, la méthode de variation des constantes en fournit une seconde $\zeta = \eta \int (e^{-\int a}/\eta^2)$, $\mathbb{C}$-linéairement indépendante de la première. Si $\eta$ est liouvillienne sur $\mathbb{C}(x)$, $\zeta$ l'est aussi, et toutes les solutions sont liouvilliennes (sur $\mathbb{C}(x)$). Enfin, le changement de variable $z = e^{\frac{1}{2}\int a} y$ transforme $(E)$ en $(E')$ $z'' + (b - \frac{1}{4}a^2 - \frac{1}{2}a')z = 0$, et respecte la propriété d'être liouvillien sur $\mathbb{C}(x)$. Nous supposerons donc dorénavant que l'EDOLH du second degré considérée est de la forme :

$$y'' = ry, \quad r \in \mathbb{C}(x), r \neq 0, \qquad (D)$$

et que $\eta, \xi$ forment une base des solutions de $(D)$ (dans une extension différentielle de $\mathbb{C}(x)$, unique à isomorphisme différentiel près). Soit alors $K = \mathbb{C}(x)\langle \eta, \xi \rangle$ l'extension de Picard-Vessiot de $\mathbb{C}(x)$ associée à $(D)$. Soit $W = \begin{vmatrix} \eta & \xi \\ \eta' & \xi' \end{vmatrix} \in K$ le wronskien de $\eta$ et $\xi$. Un calcul montre que $W' = 0$ et que $\sigma(W) = (dét(\rho(\sigma)))W$, où $\rho : \mathrm{Gal}(K/\mathbb{C}(x)) \to GL_2(\mathbb{C})$ est la représentation associée à la base $(\eta, \xi)$ de solutions de $(D)$. Ces deux égalités entraînent l'inclusion $\rho(\mathrm{Gal}(K/\mathbb{C}(x))) \subset SL_2(\mathbb{C})$ et la classification des sous-groupes fournit les différents cas suivants pour $(D)$ :

**Théorème 3** [6] *Soit $G$ le groupe de Galois de l'extension $K$ associée à $(D)$ :*

1. *si $G$ est triangularisable, l'équation $(D)$ admet une solution de la forme $e^{\int \omega}$, $\omega \in \mathbb{C}(x)$;*
2. *si $G$ est conjugué à un sous-groupe de $D^+$ et n'est pas triangularisable, l'équation $(D)$ admet une solution de la forme $e^{\int \omega}$, où $\omega$ est algébrique de degré 2 sur $\mathbb{C}(x)$;*
3. *si $G$ est fini et n'est ni triangularisable ni conjugué à un sous-groupe de $D^+$, l'équation $(D)$ possède toutes ses solutions algébriques sur $\mathbb{C}(x)$;*
4. *sinon $G = SL_2(\mathbb{C})$ et aucune solution de $(D)$ n'est liouvillienne.*

Les références pour les groupes algébriques sont [1, 9, 3], pour l'algèbre différentielle [4, 5]. Voir aussi les deux volumes [10, 8], orientés calcul formel, et [7] pour la théorie de Galois différentielle.

## 2 L'algorithme de Kovačić

Afin de résoudre l'équation $(D)$, l'algorithme de Kovačić examine successivement les cas 1-3 ci-dessus. Dans chaque cas, la solution est recherchée sous la forme $\eta = e^{\int \omega}$, le comportement des pôles de la fraction rationnelle $r$ fournissant une partie du développement fractionnaire de la dérivée logarithmique $\omega$ de $\eta$, le comportement de $\omega$ en ses autres pôles étant déterminé par un polynôme satisfaisant une équation différentielle déduite de $(D)$, et de degré défini par l'ordre des pôles de $r$. L'observation que $(D)$ possède toutes ses solutions liouvilliennes si et seulement si elle en possède une, jointe à la non-résolubilité de $SL_2(\mathbb{C})$, établit la complétude de l'algorithme. Celui-ci reposant sur l'analyse des pôles de cette fonction, la proposition suivante serre au plus près leur comportement dans chacun des cas. Si $r = P/Q$, $P, Q \in \mathbb{C}[x]$, l'ordre de $r$ à l'infini est par définition $deg(Q) - deg(P)$.

**Proposition 4** *Soit $G = \mathrm{Gal}(K/\mathbb{C}(x))$ le groupe de Galois de l'équation $(D)$ :*

1. *si $G$ est triangularisable, chaque pôle de $r$ est soit d'ordre 1 soit d'ordre pair. L'ordre de $r$ à l'infini est soit pair, soit plus grand que 2;*
2. *si $G$ est conjugué à un sous-groupe non triangularisable de $D^+$, la fraction rationnelle $r$ possède au moins un pôle qui est soit d'ordre 2, soit d'ordre impair supérieur à 2.*
3. *si $G$ est fini et 1, 2 sont exclus, alors l'ordre d'un pôle de $r$ est inférieur ou égal à 2, et son ordre à l'infini est au moins égal à 2.*

Le cas 1 est établi par examen du comportement des séries de Laurent de $r$ et de la solution cherchée $\eta$ en chaque pôle de $r$, ainsi qu'à l'infini. Le cas 2 examine l'équation différentielle déduite de $(D)$ satisfaite par la dérivée logarithmique de l'invariant $(\eta\xi)^2$ du groupe de Galois. Enfin le cas 3 examine le comportement des séries de Puiseux des solutions de $(D)$ en tout point de $\mathbb{C}$.
Nous détaillons à présent le cas 1 de l'algorithme : il est représentatif des deux autres cas, qui diffèrent toutefois de façon essentielle par leurs preuves de correction. Chaque cas repose sur les conditions nécessaires introduites ci-dessus. Soit $\Gamma$ l'ensemble des pôles de $r$.

- Si $c \in \Gamma$ est d'ordre 1, on pose $\theta_c = 0$, $\alpha_c^+ = \alpha_c^- = 1$.
- Si $c \in \Gamma$ est d'ordre 2, on pose $\theta_c = 0$. Soit $a$ le coefficient de $1/(x-c)^2$ dans la série de Laurent de $r$ en $c$, on pose $\alpha_c^\pm = \frac{1}{2}(1 \pm \sqrt{1+4a})$.

- Si $c \in \Gamma$ est d'ordre $2n$, $n \geq 2$, $\theta_c$ est la somme des termes de la forme $b_i/(x-c)^i$, $i = 2, \ldots, n$, dans la série de Laurent de $\sqrt{r}$ en $c$. Soit $a$ le coefficient de $1/(x-c)^{n+1}$ dans la série de Laurent de $r$ en $c$, diminué du coefficient de $1/(x-c)^{n+1}$ dans $\theta_c^2$, et soit $b$ celui de $1/(x-c)^n$ dans $\theta_c$. On pose $\alpha_c^{\pm} = \frac{1}{2}(\pm\frac{a}{b} + n)$.
- Si l'ordre de $r$ à l'infini est 2, alors $\theta_\infty = 0$. Soit $a$ le coefficient de $1/x^2$ dans la série de Laurent de $r$ a l'infini, on pose $\alpha_\infty^{\pm} = \frac{1}{2}(1 \pm \sqrt{1+4a})$.
- Si l'ordre de $r$ à l'infini est $-2n \leq 0$, $\theta_\infty$ est la somme des termes $b_i x^i$, $i = 0, \ldots, n$, dans la série de Laurent de $\sqrt{r}$ à l'infini. Soit $a$ le coefficient de $x^{n-1}$ dans la série de Laurent de $r$ à l'infini, diminué de celui de $x^{n-1}$ dans $\theta_\infty^2$, et $b$ celui de $x^n$ dans $\theta_\infty$. On pose $\alpha_\infty^{\pm} = \frac{1}{2}(\pm\frac{a}{b}+n)$.
- Une suite $s = (\epsilon_c)_{c \in \Gamma \cup \{\infty\}}$ telle que $d = \alpha_\infty^{\epsilon_\infty} - \sum_\Gamma \alpha_c^{\epsilon_c} \in \mathbb{N}$ définit une fraction rationnelle :

$$\theta = \sum_\Gamma (\epsilon_c \theta_c + \frac{\alpha_c^{\epsilon_c}}{x-c}) + \epsilon_\infty \theta_\infty.$$

- Pour chaque fraction $\theta$ ainsi obtenue, on recherche un polynôme unitaire $P$ de degré $d$ tel que $P'' + 2\theta P' + (\theta' + \theta^2 - r)P = 0$.

Si $\theta$ et $P$ ainsi définis existent, $\eta = Pe^{\int \theta} = e^{\int (\frac{P'}{P} + \theta)}$ est solution de $(D)$, sinon le groupe de Galois de $(D)$ n'est pas triangularisable. La preuve de correction de cette étape repose sur l'équation de Riccati associée à $(D)$ : $(R)\quad \omega' + \omega^2 = r$. Si $\eta = e^{\int \omega}$, la fonction $\omega$ est solution de $(R)$ si et seulement si $\eta$ est solution de $(D)$. Comme pour la recherche des conditions nécessaires, il suffit d'examiner les séries de Laurent de $\omega$ et de $r$.
Le cas 2 repose sur l'invariant $(\eta\zeta)^2 \in \mathbb{C}(x)$, et sur l'équation différentielle satisfaite par sa dérivée logarithmique en conséquence de $(D)$. Enfin, le cas 3 repose sur l'existence des invariants suivants du groupe de Galois de $(D)$ (Fuchs, Klein) :

- Cas tétraédrique, $T = \eta^4 + 8\eta\zeta^3$, $T^3 \in \mathbb{C}(x)$, on pose $n = 4$.
- Cas octaédrique, $O = \eta^5\zeta - \eta\zeta^5$, $O^2 \in \mathbb{C}(x)$, on pose $n = 6$.
- Cas icosaédrique, $I = \eta^{11}\zeta - 11\eta^6\zeta^6 - \eta\zeta^{11}$, $I \in \mathbb{C}(x)$, on pose $n = 12$.

Ceci s'établit en factorisant ces polynômes dans $\mathbb{Q}(\xi)$, $\xi$ racine de l'unité adéquate, et en utilisant la représentation de ces groupes définie plus haut. Si $\eta$ est algébrique, sa dérivée logarithmique $\omega$ l'est aussi. Le degré de cette dernière est minoré par $n$, par considérations des sous-groupes cycliques de $A_4$, $S_4$ et $A_5$. De plus, si $\omega$ est suppose fixé par les sous-groupes cycliques engendrés par $A(\xi)$, ce qui est possible par conjugaison, $\omega$ est exactement de degré $n$. La fin de la preuve consiste à trouver une équation différentielle satisfaite par le polynôme minimal de $\omega$.
Pour cela, on introduit le système différentiel sur $\mathcal{M}(\mathbb{C})$ suivant, où $z \in \mathcal{M}(\mathbb{C})$, $r \in \mathbb{C}(x)$ étant le coefficient rationnel de $(D)$ :

$$(E_n) \quad \left\{ \begin{array}{rcl} a_n & = & -1, \\ a_{i-1} & = & -a_i' - z a_i - (n-i)(i+1) a_{i+1} r, \quad i = n, \ldots, 0, \ z \in \mathcal{M}(\mathbb{C}), \end{array} \right.$$

et le polynôme associé $P_n = \sum_{i=0}^n \frac{a_i}{(n-i)!} w^i \in \mathcal{M}(\mathbb{C})[w]$. Par une solution de $(E_n)$, on entend une fonction méromorphe $z \in \mathcal{M}(\mathbb{C})$ telle que $a_{-1} = 0$. Si $z$ est solution de $(E_n)$, alors toute racine $\theta \in \mathcal{M}(\mathbb{C})$ de $P_n$ donne une solution $e^{\int \theta}$ de $(D)$. Si de plus $z \in \mathbb{C}(x)$, alors $a_i \in \mathbb{C}(x)$,

$i = n, \ldots, 1$, et $\theta$ est algébrique sur $\mathbb{C}(x)$. On prouve réciproquement que si $\theta$ algébrique de degré $n$ est dérivée logarithmique d'une solution de $(D)$, alors son polynôme minimal est de la forme $P_n$, et le coefficient de $x^{n-1}$ dans $P_n$ fournit une solution rationnelle de $(E_n)$. Mais alors, compte tenu des bornes sur les degrés des dérivées logarithmiques des solutions de $(D)$, lorsque $(E_4)$ (resp. $(E_6)$) possède une solution rationnelle, le polynôme $P_4$ (resp. $P_6$) est irréductible sur $\mathbb{C}(x)$. Ceci est également vrai de $(E_{12})$ et $P_{12}$, lorsque $(E_4)$ et $(E_6)$ n'ont pas de solutions rationnelles. En fait, le système $(E_n)$ exprime la nullité des coefficients du polynôme $B \in \mathbb{C}(x)[w]$, $B = \frac{1}{X^n u}(X^n u A(Y,X))'$, lorsque $w = X/Y$, $u = e^{\int a_{n-1}}$, $A(w)$ polynôme minimal de $\theta = \frac{\nu'}{\nu}$, $\nu$ solution algébrique de $(D)$, et $A(X,Y)$ est l'homogénéisé de $A \in \mathbb{C}(x)[w]$. En effet, $A$ et $B$ s'annulent tous deux en $\theta$, ce qui implique $B = 0$, $A$ étant minimal. La dernière observation est que toute forme homogène $F$ de degré $n$ en les solutions de $(D)$, algébriques ou non, a sa dérivée logarithmique solution de $(E_n)$. C'est là un exercice classique sur les fonctions symétriques : on se ramène par linéarité à $F = \prod_{i=1}^{n} \eta_i$, puis, pour $1 \leq m \leq n$, on définit :

$$\begin{aligned} \sigma_{m,k} &= 0 \quad \text{si } k < 0 \text{ ou } k > m, \qquad \sigma_{m,1} = 1, \\ \sigma_{m,k} &= \textstyle\sum_{1 \leq i_1 < \cdots < i_k \leq m} \omega_{i_1} \cdots \omega_{i_k}, \quad \omega_j = \frac{\eta_j'}{\eta_j}. \end{aligned}$$

L'identité $\sigma_{m,k} = \sigma_{m-1,k} + \sigma_{m-1,k-1}\omega_m$ donne les dérivées de ces fonctions symétriques :

$$\sigma_{m,k}' = (m+1-k)r\sigma_{m,k-1} - \sigma_{m,1}\sigma_{m,k} + (k+1)\sigma_{m,k+1},$$

qui permettent d'établir par induction que :

$$a_i = (-1)^{n-i+1}(n-i)!\sigma_{n,n-1}, \quad a_i \text{ coefficient de } (E_n).$$

D'où $a_{n-1} = \frac{F'}{F}$ et $a_{-1} = 0$. Il suffit alors de prendre pour $F$ les invariants rationnels $T^3$, $O^2$ et $I$. Comme pour les cas 1 et 2, la comparaison des séries de Laurent de $r$ et $F'/F$, via $(E_n)$, $F$ l'un des semi-invariants $T$, $O$ ou $I$, permet de calculer la partie fractionnaire de $\omega$ relative aux pôles de $r$, la partie fractionnaire manquante étant déduite de $(E_n)$, via une équation différentielle linéaire dont on recherche une solution polynomiale, de degré donné par les premiers coefficients des séries de Laurent de $r$ en ses pôles et à l'infini.

## Références


[1] A. Borel. *Linear Algebraic Groups*, volume 126 of *Graduate Text in Mathematics*. Springer-Verlag, 1991.

[2] A. Duval and M. Loday-Richaud. Kovačić's algorithm and application to some special functions. *Applicable Algebra in Engineering, Communication and Computing*, 3(3), 1992. To appear.

[3] J. E. Humphreys. *Linear Algebraic Groups*, volume 21 of *Graduate Text in Mathematics*. Springer-Verlag, 1981.

[4] I. Kaplansky. *An Introduction to Differential Algebra*. Hermann, 1957.

[5] E. R. Kolchin. *Differential Algebra and Algebraic Groups*, volume 54 of *Pure and Applied Math.* Academic Press, New York, 1973.

[6] J. J. Kovačić. An Algorithm for Solving Second Order Linear Homogeneous Differential Equations. *Journal of Symbolic Computation*, 2:3–43, 1986.

[7] A. H. M. Levelt. Differential Galois theory and tensor products. *Indagationes Mathematicae*, 1(4):439–450, 1990.

[8] M. F. Singer, editor. *Differential Equations and Computer Algebra*, Computational Mathematics and Applications. Academic Press, 1991.

[9] T. A. Springer. *Linear Algebraic Groups*, volume 9 of *Progress in Math.* Birkhäuser, 1981.

[10] E. Tournier, editor. *Computer Algebra and Differential Equations*, Computational Mathematics and Applications. Academic Press, 1989.

# 12

# Functions in Symbolic Computation: Time to Move On

John R. Shackell
University of Kent, Canterbury

[summary by Kevin J. Compton]

The existing approach to symbolic computation is based on elementary functions (built from 1, $\pi$, and variables like $x$ using $+$, $-$, $\times$, $\div$, and functions exp, log, sin). Standard notation gives a way of representing such functions. Symbolic computation with elementary functions is highly developed.

- To test *zero equivalence* one uses the Risch-Norman method plus the assumption of Schanuel's conjecture to distinguish constants.
- *Differentiation* of elementary functions is easy; integration is hard, but much has been achieved.
- *Differential equations* are harder still, but there are some results.
- Determining *limits* is undecidable if unrestricted use of sine is allowed. (See Shackell [5].) However, if sine is excluded there is a theoretical basis for limits in some cases.
- For *zero isolation* (cylindrical decomposition) there is now an algorithm due to Richardson [1] but as yet no implementation.

There are a number of gaps to be filled but symbolic computation must still be reckoned a success. Nonetheless, Shackell believes that it is time for the main thrust of research in this area to start along different lines. There are several reasons for this.
First, the answer to a problem may not be given by an elementary function. In some cases we do not know whether the answer is elementary or not. This is a hard question. If it isn't elementary (or at least Liouvillian, see below) it cannot be expressed in standard systems for representing functions. Usually, a scientist or engineer is just concerned with getting an answer, not in beautiful mathematics: a proof that an elementary solution does not exist is not of interest.
Second, the system should provide qualitative answers. Stoutemyer has argued that the user needs help to interpret a complicated formula. The system should give information about zeros, singularities, limits, symmetries, and perhaps sketch a graph.
This leads to the question of whether we really need a formula at all. If not, do functions have to be elementary? The difficulty is that we need algorithms so we must have some type of representation.
**Main Thesis.** *Shackell suggests taking an algebraic differential equation approach. In this approach functions are built up using algebraic differential equations.*
For example, one might specify

$$f(x) = x^{-2}a_1'(x) + (x+1)a_2(x) + x^3,$$

where $a_2$ satisfies $y''y+(x+a_1(x))y'^2+x^2+3=0$, $y(0)=1$, $y'(0)=2$, and $a_1$ satisfies $y'''=x^3y'^2+y''$, $y(0)=0$, $y'(0)=2$, $y''(0)=3$. Thus, in this approach there is a tower of differential fields

$$\mathbb{R}(x)=\mathcal{F}_0\subset\mathcal{F}_1\subset\cdots\subset\mathcal{F}_n$$

such that $\mathcal{F}_{i+1}$ is a differential extension of $\mathcal{F}_i$ by a solution of an algebraic differential equation over $\mathcal{F}_i$. (In fact, any function in $\mathcal{F}_n$ satisfies an algebraic differential equation over $\mathbb{R}$, but there are advantages in using the tower).

**Definition 1** *The functions obtained by taking all differential equations of the form $y'=r_i(y)$ with $r_i$ a rational function over $\mathcal{F}_i$ are called* Pfaffian functions.

**Definition 2** *The functions obtained by taking all differential equations either of degree 0 (algebraic extension) or of the form $y'=yf_i$ with $f_i\in\mathcal{F}_i$ (exponential extension) are called* Liouvillian functions.

Are these two classes of functions reasonable for symbolic computation?

- For *zero equivalence* there are algorithms modulo an oracle for constants. (See Shackell [6] for the Pfaffian case and general case.)
- *Integrals* and solutions of *differential equations* exist within the system automatically.
- *Zero isolation* can be done in the Pfaffian case and slightly beyond. (See Richardson [1].)
- Perhaps symmetries can be computed. The problem is reducible to zero isolation, so can be done in the Pfaffian case. This may work in a more general case.
- Limits in the Liouvillian case are more or less done modulo constants [3, 4]. Perhaps this is extendable to the Pfaffian case. There is some information in the more general case, but it is undecidable in full generality.
- Should symbolic integration be ignored? The integral of $e^x$ should not be defined as the solution of the differential equation $y'=e^x$. The functions provided by the user (corresponding to the tower of differential fields) may have physical significance. It would be nice to give the solution as simply as possible in terms of these. Mike Singer [7] has done some relevant work for linear differential equations, but the problem is probably very hard in general.

Everything so far has been modulo a method for deciding the signs of constants. The Schanuel conjecture gives a method for elementary functions. What might we do in the more general case? Shackell suggests taking a numeric/symbolic approach in which we handle constants numerically but otherwise work symbolically. Algorithms would need modification to take account of approximation of coefficients. Answers would only be correct modulo a range of error and might need special interpretation. Partial answers may have to be accepted, but there may be no alternative. In practice one needs a fast method of determining the sign of a constant. Perhaps even where the purely symbolic approach is possible, the numeric/symbolic approach would be faster.
One might expect to get answers of the following type. For symmetry one might have that $f(a+x)=f(a-x)\pm\varepsilon g(x)$. For limits, expressions like $f_0(x)\sim(1\pm\varepsilon)e^x$ and $f_1(x)\sim e^{(1\pm\epsilon)x}$ should be acceptable, but not $f_2(x)\sim 2x\pm\varepsilon^x$. (The asymptotic expression for $f_2$ might even be acceptable in some ranges.)

A purely symbolic approach for handling constants looks hard in general. The Schanuel conjecture says roughly that the relations between constants are the ones we already know. We know little about Pfaffian constants. New work by Richardson [2] gives some glimmers of hope but may not work out.
Shackell suggests that the future work is needed in the following areas.

- Solve the constants problem using a numeric/symbolic approach if necessary.
- Extend word on Liouvillian case to Pfaffian case.
- Check out the method for finding symmetries.
- Gain practical experience through pilot implementations.

What level of generality should we have in practical development? Shackell compares three alternatives: Liouvillian functions, Pfaffian functions, and arbitrary towers of algebraic differential equations. The first is not very adventurous but still might be worthwhile. The problem with the last is the scarcity of algorithms; there are only a zero equivalence modulo constants, a little on limits, and a little on symmetry. Pfaffian functions seem to offer the most hope. There are quite a number of algorithms for this class, there is some chance of generalizing the limit algorithm, and there is a better chance of finding a method to deal with constants than in the case of towers of algebraic differential equations. The biggest problem is that many functions that arise will not be Pfaffian.
In conclusion, the range of functions that can be handled by algebra systems is very important and there is a good chance for development here.

## References

[1] D. Richardson. Computing (in a bounded part of the plane) the topology of a real curve defined by functions which satisfy first order algebraic differential equations. Preprint.

[2] D. Richardson. The elementary constant problem. Preprint.

[3] J. R. Shackell. Limits of Liouvillian functions. Preprint.

[4] J. R. Shackell. Limits of Liouvillian functions II: extensions via analytic functions. Preprint.

[5] J. R. Shackell. Asymptotic estimation of oscillating functions using an interval calculus. In Springer-Verlag, editor, *Symbolic and Algebraic Computation*, pages 481–489, New York, 1989.

[6] J. R. Shackell. Zero equivalence in function fields defined by algebraic differential equations. *Transactions of the American Mathematical Society*, 1992. To appear.

[7] M. F. Singer. Liouvillian solutions of linear differential equations with Liouvillian coefficients. *Journal of Symbolic Computation*, 11:251–273, 1991.

# 13

# Function Composition and Automatic Average-Case Analysis

Paul Zimmermann
INRIA, Rocquencourt

[summary by Paul Zimmermann]

This talk introduces the composition of functions defined over extended context-free languages. It is shown that this composition is automatically computable. It enables the automatic analysis of complex problems with *small* input descriptions, for example repeated differentiation or iterated automata on regular languages.

In the field of automatic complexity analysis, the length of the problem description is often a limitation: writing a long specification program is often a difficult error prone process. Thus one needs some powerful constructs to describe algorithms, with the necessary constraint that these constructs allow an *automatic* analysis.
One is interested here in the average case analysis of programs including some *compositions* of functions. None of the existing systems, including METRIC [5], COMPLEXA [10] or the version of Lambda-Upsilon-Omega ($\Lambda\Upsilon\Omega$) described in [2], is able to analyze the composition of functions. The main reason may be the following: in these systems, the analysis of statements like $f(x)$ relies on the fact that all required data types are defined, either implicitly like in METRIC and COMPLEXA where all data structures are lists, or explicitly like in Lambda-Upsilon-Omega. But in the statement $f(g(y))$, the difficulty is to get a formal description of the object $g(y)$, which is not known *a priori*.
As an example, suppose one has written a function *diff* performing the differentiation of symbolic expressions with respect to one variable, and now one would like to analyze the two-fold differentiation by just defining

$$diff2(e) \stackrel{\text{def}}{=} diff(diff(e))$$

instead of having to write the entire body of the function *diff2*. P. Zimmermann shows that this shorthand is possible: he defines a class of programs including function compositions, such that every program can be *automatically* expanded into another one without any composition, and equivalent to the original one in what concerns complexity analysis. This result allows us to define and to analyze large problems by short description programs.

## 1 A class of programs with composition

This section introduces the composition of functions in the ADL language (Algorithm Description Language), especially designed for automatic average case analysis in the Lambda-Upsilon-Omega system [1, 2, 7]. The following is an ADL program performing the differentiation of symbolic expressions.

```
type expression = zero | one | x
                | plus(expression,expression)
                | times(expression,expression);
     plus,times,zero,one,x = atom(1);

function diff(e : expression) : expression;
begin
   case e of
      plus(e1,e2)          : plus(diff(e1),diff(e2));
      times(e1,e2)         : plus(times(diff(e1),copy(e2)),
                                  times(copy(e1),diff(e2)));
      zero                 : zero;
      one                  : zero;
      x                    : one
   end;
end;
```

where the function `copy`, which simply makes a carbon copy of one expression, is also defined in the same manner. To define the complexity measure as the number of atoms in the output of *diff*, it suffices to define the cost of each atom as 1:

```
measure plus,times,zero,one,x : 1;
```

If one analyzes the *diff* function in the Lambda-Upsilon-Omega system, one gets the following average cost for expressions of size $n$:

$$\tau\overline{diff}_n = \frac{1}{4}\sqrt{2\pi}n^{3/2} + O(n).$$

Now one allows the use of the composition in ADL programs, that is statements of the form $f(g(y))$ where $f$ and $g$ are two functions defined in the program, and $y$ is a local variable. For example, the second order differentiation is defined as follows

```
function diff2(e : expression) : expression;
begin
   diff(diff(e))
end;
```

**Definition 1** *The* composition graph *associated to an* ADL *program is the graph whose vertices are the function names, and for each composition $f(g(\ldots))$ in the body of a function $h$, there is an arrow from $h$ to all functions on which $f$ and $g$ depend (the relation "depends on" is the reflexive and transitive closure of the relation "has in its body").*

**Theorem 1** *If the composition graph of an* ADL *program is acyclic, then the program translates into an equivalent program without composition.*

The proof of this theorem involves the definition of a way of expanding the composition, the *expansion process*. This process necessarily terminates when the composition graph is acyclic.
As the average case analysis of ADL programs *without* composition is already known to be automatic [2, 7], the above theorem implies directly the following result:

**Corollary 1** *The average case analysis of* ADL *programs with an acyclic composition graph is automatic.*

# 2 Two non-trivial examples

This section presents two research problems where the implementation of the expansion process on a computer allowed P. Zimmermann to discover some results which would have been very difficult to find by hand.

## 2.1 Analysis of $k$th order differentiation

The expansion process has been encoded in the version V1.4 of the system Lambda-Upsilon-Omega. When one analyzes the function `diff2` as `diff o diff`, the system displays with "printlevel" 3 the expanded form of the function body:

```
function diff_of_diff (e : expression) : expression;
begin
   case e of
      (plus,(e1,e2)) : plus(diff_of_diff(e1),diff_of_diff(e2));
      (times,(e1,e2)) : plus(plus(times(diff_of_diff(e1),copy_of_copy(e2)),
                                  times(copy_of_diff(e1),diff_of_copy(e2))),
                             plus(times(diff_of_copy(e1),copy_of_diff(e2)),
                                  times(copy_of_copy(e1),diff_of_diff(e2))));
      zero : zero;
      one : zero;
      x : zero;
   end;
end;
```

Three other new functions have been also introduced, namely `diff_of_copy`, `copy_of_diff` and `copy_of_copy` (the function `copy` is not initially known by the system). The system then proceeds in the usual way (Algebraic Analysis, Solver, Analytic Analysis) described in [2] and gives the final result:

```
Average cost for diff2 on random inputs of size n is:

                                    2        3/2
                                 (1/2 n ) + (O(n   ))

for n mod 2 = 1, and 0 otherwise.
```

In this way, by just adding each time one more call to the function `diff`, P. Zimmermann was able to analyze the $k$-fold iterated differentiation until $k = 7$, and obtained the following figures.

| | average cost | average cost | |
|---|---|---|---|
| `diff` | $\frac{1}{4}\sqrt{2\pi}n^{3/2} + O(n)$ | `diff2` | $\frac{1}{2}n^2 + O(n^{3/2})$ |
| `diff3` | $\frac{3}{16}\sqrt{2\pi}n^{5/2} + O(n^2)$ | `diff4` | $\frac{1}{2}n^3 + O(n^{5/2})$ |
| `diff5` | $\frac{15}{64}\sqrt{2\pi}n^{7/2} + O(n^3)$ | `diff6` | $\frac{3}{4}n^4 + O(n^{7/2})$ |
| `diff7` | $\frac{105}{256}\sqrt{2\pi}n^{9/2} + O(n^4)$ | | |

These figures led to conjecture an average cost of

$$\frac{\Gamma(k/2+1)}{2^{k/2}}n^{k/2+1} + O(n^{(k+1)/2}) \tag{1}$$

for the $k$th order differentiation, that was proved to be correct afterwards.

## 2.2 Regular languages and the Collatz conjecture

This example shows that function composition, used jointly with analysis of functions with a finite number of return values [8], helps to compute grammars of sets derived from regular languages. The Collatz conjecture says: "starting from a positive integer, the iteration of the function

$$f(n) = \begin{cases} n/2 & \text{if } n \text{ is even,} \\ 3n+1 & \text{if } n \text{ is odd,} \end{cases}$$

ultimately reaches 1". For example, one obtains the following chain for the number 13: $13 \to 40 \to 20 \to 10 \to 5 \to 16 \to 8 \to 4 \to 2 \to 1$. In [6], David Wilson introduced the sets $S_k$, where the index $k$ denotes the number of times the function $3n+1$ is applied before 1 is reached. In the above example, the function $3n+1$ is applied two times (from 13 to 40 and from 5 to 16), thus 13 belongs to $S_2$. The first sets begin like this:

$$\begin{aligned} S_0 &= \{1,2,4,8,16,32,64,128,256,512,\ldots\} \\ S_1 &= \{5,10,20,21,40,42,80,84,85,160,\ldots\} \\ S_2 &= \{3,6,12,13,24,26,48,52,53,96,\ldots\} \\ S_3 &= \{17,34,35,68,69,70,75,136,138,140,\ldots\}. \end{aligned}$$

Wilson and Shallit proved[4] in [4] that sets $S_k$ are 2-automatic, that is the base-two string expressions of each $S_k$ form a regular language (accepted by a finite automaton and writable as a regular expression). For instance, $S_0 \to 1\ 0^*$ and $S_1 \to 101\ (01)^*\ 0^*$. This result implies that the number of $n$-bit integers in $S_k$ is easily computable: it is the coefficient of $z^n$ in a rational function derived from the regular expression of $S_k$, for example $z^3/(1-z^2)/(1-z)$ for $S_1$.
Function composition enables one to compute a grammar for $S_k$ *automatically*, with a description file of *linear* length with respect to $k$. From this grammar, one easily derives a regular expression. At the end of this section, such regular expressions for $S_2$ and $S_3$ are given. The idea is to define the function $g$ dividing its input by two as long as possible, then applying *one time* the function $3n+1$:

$$g(n) = \begin{cases} g(n/2) & \text{if } n \text{ is even,} \\ 0 & \text{if } n = 1, \\ 3n+1 & \text{otherwise.} \end{cases}$$

For instance, $g(13) = 40$, $g(40) = 16$ and $g(16) = 0$; the function $g$ gives a characterization of $S_k$:

$$S_k = \{n \mid g^{(k)}(n) \text{ is a power of two }\}. \tag{2}$$

Therefore to construct an ADL program recognizing integers in $S_k$, one has to encode the function $g$, and a function recognizing powers of two. For this purpose, integers are encoded in base two:

```
type integer = nil | bit integer;
     bit = zero | one;
     zero, one = atom(1);
     nil = atom(0);
```

The function $g$ is written using a function called `three_x_plus_1`, whose input is the base-two representation of an integer $n$, and which outputs the base-two representation of $3n+1$:

[4] This result could also be deduced from the theory of *sequential transductors* [3, example 8 page 123].

```
function three_x_plus_1 (i : integer) : integer;
begin
    case i of
        nil : product(one,nil);
        (zero,j) : product(one,three_x(j));
        (one,j) : product(zero,three_x_plus_2(j));
    end;
end;
```

The other functions `three_x` and `three_x_plus_2` are defined similarly. With the functions `g` and `is_a_power_of_two`, according to equation (2), one writes the function `is_in_S3` to recognize integers in $S_3$:

```
function g (i : integer) : integer;
begin
    case i of
        nil : nil;
        (zero,j) : g(j);
        (one,nil) : nil;
        otherwise : three_x_plus_1(i);
    end
end;

function is_a_power_of_two (i : integer) : boolean;
begin
    case i of
        nil : false;
        (zero,j) : is_a_power_of_two(j);
        (one,j) : is_zero(j)
    end;
end;

function is_in_S3 (i : integer) : boolean;
begin
    is_a_power_of_two(g(g(g(i))))
end;
```

During the analysis of this whole program, the $\Lambda\Upsilon\Omega$ system prints some messages, for example (among other lines):

```
Computing composition of is_a_power_of_two and g : f2
Computing composition of f2 and g : f8
Computing composition of f8 and g : f26
Introducing the new type T84 for which function f26 returns true
Introducing the new type T142 for which function f26 returns false
```

At this stage, it has constructed a set of ADL functions without any composition, containing the function `f26` equivalent to `is_in_S3`. For such a set, it is possible to derive automatically a grammar of the data structures for which each function with a finite number of possible outputs (in particular a boolean function like `f26`) returns a given value [8]. For example, as explained by the last lines in the above messages, the system introduced two new data types `T84` and `T142`, which stand for the integers in $S_3$ and not in $S_3$ respectively. Like for the expansion process, a complete grammar for `T84` and `T142` was in fact generated, starting from the grammar of the type `integer`.

Due to the form of the rules used (cf [8]), this grammar is unambiguous because so was the grammar of `integer`. The raw grammar one gets has 58 non-terminals, among them 27 do not derive any finite string. After some simplifications by hand (they took longer than the automatic construction of the grammar!), P. Zimmermann got the following regular expression for $S_3$:

$$\begin{aligned} S_3 \quad \rightarrow \quad & ((\epsilon \mid (100101111011010000)^*1001011\ (\epsilon \mid 1 \mid 1101 \mid 110110011 \mid 11011010000 \mid 11011010000011)) \\ & (100011)^*1000 \mid (100101111011010000)^*100101\ (\epsilon \mid 11101100 \mid 1110110100000))(\epsilon \mid 1)\ (10)^*10^*. \end{aligned}$$

Similarly, he computed with the help of the Lambda-Upsilon-Omega system the following regular expression of the set $S_2$, starting from a grammar with 22 non-terminals:

$$S2 \rightarrow (1 \mid 11100\ (011100)^*\ (0 \mid 01))\ (10)^*\ 1\ 0^*$$

**Conclusion.** This research shows that some kinds of function compositions are well suited for an automatic average case analysis. The main idea is the following: a program including compositions first translates into an similar program without composition by an expansion process, then this last program is analyzed by already known techniques [2].
Composition of functions is not only useful in the description of algorithms, but in some cases it is *necessary* to use it, otherwise the description would be too long, as the two examples presented here prove it. In these cases, the long description is automatically generated by the computer, therefore it contains no error.

## References


[1] P. Flajolet, B. Salvy, and P. Zimmermann. Lambda–Upsilon–Omega: The 1989 Cookbook. Rapport de recherche 1073, Institut National de Recherche en Informatique et en Automatique, August 1989. 116 pages.

[2] P. Flajolet, B. Salvy, and P. Zimmermann. Automatic Average–case Analysis of Algorithms. *Theoretical Computer Science*, 79(1):37–109, February 1991.

[3] J. E. Pin. *Variétés de langages formels*. Études et recherches en informatique. Masson, 1984.

[4] J. Shallit and D. Wilson. The "$3x+1$" Problem and Finite Automata. *Bulletin of the EATCS*, 46:182–185, 1992.

[5] B. Wegbreit. Mechanical Program Analysis. *Communications of the ACM*, 18(9):528–539, September 1975.

[6] D. W. Wilson. Transaction ⟨1778@cvbnetPrime.COM⟩ of usenet.sci.math, 1991.

[7] P. Zimmermann. *Séries génératrices et analyse automatique d'algorithmes*. Thèse de doctorat, École Polytechnique, Palaiseau, 1991.

[8] P. Zimmermann. Analysis of functions with a finite number of return values. Research Report 1625, Institut National de Recherche en Informatique et en Automatique, February 1992.

[9] P. Zimmermann. Function composition and automatic average case analysis. In P. Leroux and C. Reutenauer, editors, *Séries formelles et combinatoire algébrique*, volume 11 of *Publications du LACIM, Université du Quebec à Montréal*, pages 477–486, 1992. Proceedings of the 4th Colloquium. To appear in *Discrete Mathematics*.

[10] W. Zimmermann. *Automatische Komplexitätsanalyse funktionaler Programme*. PhD thesis, Fakultät für Informatik der Universität Karlsruhe, June 1990. Also available in the collection *Informatik Fachberichte*, number 261, Springer Verlag.

# Part III

# Asymptotic Analysis

# 14

# Théorèmes taubériens pour l'énumération asymptotique

Kevin Compton
University of Michigan and INRIA, Rocquencourt

[résumé par Paul Zimmermann]

Dans son livre intitulé *Divergent Series* [2], G.H. Hardy définit les théorèmes Taubériens à partir des théorèmes Abéliens :

> *"An 'Abelian' theorem is, roughly, one which asserts that, if a sequence or function behaves regularly then some average of the sequence or function behaves regularly."*
>
> *"['Tauberian' theorems are] ... corrected forms of the false converses of Abelian theorems."*

Un exemple simple de cette réciprocité est le suivant :

$$(1) \qquad \lim_{n\to\infty} b_n = L$$

$$(2) \qquad \lim_{x\to 1} \frac{\sum_{n=0}^{\infty} b_n x^n}{(1-x)^{-1}} = L$$

L'implication $(1) \Rightarrow (2)$ est un théorème Abélien, alors que $(2) \Rightarrow (1)$ n'est pas vrai (prendre par exemple $b_n = (-1)^n$). Mais en ajoutant une condition à (2), la réciproque devient vraie et on obtient un théorème Taubérien:

$$(2) \text{ et } b_{n+1} - b_n = o(\frac{1}{n}) \quad \Longrightarrow \quad (1).$$

On peut généraliser ce premier théorème en remarquant que le dénominateur apparaissant dans (2) est la série $\sum a_n x^n$ avec comme valeur particulière $a_n = 1$ :

> Soient les séries $a(z) = \sum a_n z^n$, $b(z) = \sum b_n z^n$ et $c(z) = b(z)/a(z)$. Si
>
> $$(i) \qquad \lim_{n\to\infty} \frac{a_{n-1}}{a_n} = R, \quad 0 < R < \infty$$
>
> $$(ii) \qquad c(z) \text{ a un rayon de convergence supérieur à } R$$
>
> alors
>
> $$\lim_{n\to\infty} \frac{b_n}{a_n} = c(R).$$

Ce dernier théorème, dû à Schur, permet par exemple de montrer que le nombre de cycles de longueur $j$ dans une permutation aléatoire suit une loi de Poisson de moyenne $1/j$. En effet, il suffit de prendre $a(z) = 1/(1-z)$, la série génératrice exponentielle des permutations, et $b(z)$ celle des permutations ayant $m$ cycles de longueur $j$, soit $(z^j/j)^m e^{-z^j/j}/(1-z)$. On montre de la même

manière que la probabilité qu'un graphe fonctionnel n'ait pas de point fixe tend asymptotiquement vers $e^{-1}$.
On peut aussi énoncer des théorèmes Taubériens plus généraux sur des fonctions au lieu de séquences, c'est-à-dire sur des transformées de Laplace au lieu de séries entières. C'est l'approche de Wiener :

> *"The merits of Wiener's method lie in its great power and generality, and the light which it throws on the whole subject; not in simplicity." G.H. Hardy*

Faisons correspondre à une séquence $(b_n)$ la fonction $f(t) = b_{\lfloor t \rfloor}$, où $\lfloor t \rfloor$ désigne la partie entière de $t$. Ainsi, on associe à la série $b(x) = \sum b_n x^n$ la fonction

$$\int_0^\infty f(t)e^{-yt}dt,$$

plus précisément $b(e^{-y}) \sim \int_0^\infty f(t)e^{-yt}dt$ lorsque $y$ tend vers 0.
Les hypothèses des théorèmes taubériens classiques sont de la forme :

$$\int_{-\infty}^\infty F(t)G(x-t)dt \overset{x\to\infty}{\longrightarrow} L\int_0^\infty G(t)dt,$$

ce que l'on abrège en $F * G \to L\|G\|_1$. Le théorème principal est le suivant.
**Théorème Taubérien de Wiener** : *Soit $\hat{G}$ la transformée de Fourier de $G$. Si pour $F \in \mathcal{L}^\infty$,*

$$\begin{array}{ll} (i) & F * G \to L\|G\|_1 \\ (ii) & \hat{G}(x) \neq 0 \text{ pour tout réel } x \end{array}$$

*alors $F * H \to L\|H\|_1$ pour toute fonction $H \in \mathcal{L}^1$.*

La plupart des théorèmes taubériens se déduisent de ce dernier en choisissant $G$ (le noyau) et $H$ convenables, par exemple la version suivante due à Pitt :

> Pour $f \in L^\infty[0,\infty)$, si l'on a $\frac{1}{x}\int_0^\infty f(t)g(\frac{x}{t})dt \to L\int_0^\infty g(t)dt$,
> et si la condition supplémentaire $\int_0^\infty g(t)t^{-ix}dt \neq 0$ est vérifiée pour tout réel $x$, alors pour tout $h$,
> $$\frac{1}{x}\int_0^\infty f(t)h(\frac{x}{t})dt \to L\int_0^\infty h(t)dt.$$

En prenant $g(t) = e^{-t}$ et $h$ la fonction indicatrice de l'intervalle $[0,1]$, on obtient le théorème de Hardy-Littlewood :

> Si $f$ bornée vérifie $\lim_{x\to\infty}\frac{1}{x}\int_0^\infty f(t)e^{-t/x}dt = L$, alors $\lim_{x\to\infty}\frac{1}{x}\int_0^\infty f(t)dt = L$.

En retournant aux séquences entières, ce théorème ne nous donne des indications que sur la moyenne de Césaro.
C'est plutôt un résultat sur la limite de $f(x)$ qui nous intéresse. Pour cela, il faut alors rajouter une condition aux hypothèses:

Si $f$ bornée *et lentement décroissante* vérifie

$$\lim_{x\to\infty} \frac{1}{x}\int_0^\infty f(t)e^{-t/x}dt = L,$$

alors $\lim_{x\to\infty} f(x) = L$.

Une fonction $f$ est dite lentement décroissante si pour tout $\epsilon > 0$, il existe $\lambda > 1$ tel que pour tout $x$ et $x \leq y \leq \lambda x$, on ait $f(y) > f(x) - \epsilon$. Ce résultat se transporte facilement dans le domaine des séquences entières:

Si $(b_n)$ est bornée et lentement décroissante, et que $\lim_{x\to 1}(1-x)\sum_0^\infty b_n x^n = L$, alors $\lim_{n\to\infty} b_n = L$.

Les fonctions à variation lente [1] donnent lieu elles aussi à des théorèmes Taubériens:

(Karamata) Si $f$ est bornée et $l$ est à variation lente avec

$$\frac{1}{x}\int_0^\infty f(t)e^{-t/x}dt \sim l(x),$$

alors $\dfrac{1}{x}\displaystyle\int_0^\infty f(t)dt \sim l(x)$.

Ce théorème se généralise en remplaçant dans l'hypothèse $1/x$ par $1/x^{\alpha+1}$, avec $\alpha > -1$, et $f(t)/t^\alpha$ bornée. Cela permet de prouver que $\frac{1}{x}\int_0^\infty f(t)/t^\alpha dt \sim l(x)$, et si en plus $f(t)/t^\alpha$ est à décroissance lente, que $f(x)/x^\alpha \sim l(x)$.
Ceci permet de montrer directement que le nombre moyen de composantes dans un graphe fonctionnel est $\sim 1/2\log x$.
En conclusion: *"Tauberian theorems should be used more widely in combinatorics. Begin with Hardy, not Feller. Feller considers only Tauberian theorems with kernel $e^{-t}$. We have found more combinatorial applications by taking $t^\alpha e^{-t}$. Hardy has a list of half a dozen other kernels. [...] prospects for combinatorial applications."*

## Références

[1] N. H. Bingham, C. M. Goldie, and J. L. Teugels. *Regular Variation.* Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1987.

[2] G. H. Hardy. *Divergent Series.* Oxford University Press, 1949.

# 15

# The Asymptotic Behaviour of Coefficients of Large Powers of Functions

Danièle Gardy
LRI, Orsay

[summary by Bruno Salvy]

The evaluation of the distribution of a sum of independent random variables, the study of asymptotic parameters of trees and forests are two examples of problems that require an asymptotic estimate of the behaviour of $[z^n]f^d(z)$ when both $n$ and $d$ tend to infinity. In this talk, D. Gardy surveys the literature and proves several new results.
The general philosophy in this domain is that the larger $d$ is with respect to $n$, the easier it is to compute an asymptotic estimate. For instance, if $n$ is fixed and $d$ tends to infinity then elementary majorisations show that $[z^n]f^d(z) \sim [z^n](f_{\alpha_0}z^{\alpha_0} + f_{\alpha_1}z^{\alpha_1})^d$, where $f_{\alpha_0}$ and $f_{\alpha_1}$ are the first non-zero coefficients of $f$. On the other end of the scale, the problem with fixed $d$ and $n$ tending to infinity is the general problem of asymptotic estimates of coefficients of generating functions which has received a lot of attention, and for which results are known in particular classes. The emphasis in this talk is on the intermediate case, when both $n$ and $d$ tend to infinity.

## 1 The saddle-point method

The saddle-point method is the method of choice in this domain. We describe it in the general case where it used to get an estimate of $[z^n]\phi(z)$, and will then comment on the implications of taking $\phi(z) = f^d(z)$ with $d$ tending to infinity.
One starts from Cauchy's theorem

$$[z^n]\phi(z) = \frac{1}{2i\pi}\oint \frac{\phi(z)}{z^{n+1}}\,dz,$$

where the contour contains the origin and no other singularity. A suitable contour is a circle through a particular point called *saddle-point*, which is a root of the derivative of the integrand (see [10] for a good intuitive justification of this):

$$\frac{\phi'(\rho)}{\phi(\rho)} = \frac{n+1}{\rho}. \tag{1}$$

If everything goes well, there is such a point on the positive real axis, it is a maximum of the integrand on the circle, locally the integrand behaves as a Gaussian and the other parts of the integral are negligible. If $\rho$ is the only maximum of $f$ on the circle of radius $\rho$, $h(z) = \log\phi(z) - (n+1)\log z$ and $h''(\rho) \neq 0$ (sufficient conditions for this will be given below), then one can see that $h(z) = h(\rho) + h''(\rho)u^2/2$ defines two functions $u(z)$ analytic at $z = \rho$ that do not vanish on

the circle. By imposing further $u'(\rho) > 0$, there is only one such function and we can change the variable in the integral, thus obtaining

$$[z^n]\phi(z) = \frac{\phi(\rho)}{2i\pi\rho^{n+1}} \int e^{h''(\rho)u^2/2} z'(u)\, du.$$

Expanding $z'(u)$ as a power series and integrating term by term from $-\infty$ to $\infty$ gives a full asymptotic expansion:

$$[z^n]\phi(z) \approx \frac{\phi(\rho)}{\rho^{n+1}\sqrt{2\pi h''(\rho)}} \left[1 + \frac{\alpha_1}{n} + \frac{\alpha_2}{n^2} + \cdots\right].$$

The computations needed by the saddle-point method are technical but not hard. The difficulty lies in the justification of its use. If one only wants the first order saddle-point estimate, obtaining this validity requires three steps: *i)* proving the existence of the saddle-point in a valid part of the complex plane, *ii)* proving that the expansion of $z'(u)$ is valid in a sufficiently large neighbourhood of the saddle-point, *iii)* bounding $h$ on the rest of the contour. A sufficient condition for *ii)* and *iii)* is that there exists a real positive function $\epsilon(n) < 1$ such that

*(a.)* $|\epsilon^2\rho^2 h''(\rho)| \to \infty, \quad n \to \infty;$

*(b.)* $|h^{(3)}(\rho e^{i\theta})| = o\left(\frac{h''(\rho)}{\rho\epsilon}\right)$, uniformly for $0 \le |\theta| \le \epsilon$.

*(c.)* $|\phi(\rho e^{i\theta})| = o\left(\frac{\phi(\rho)}{\rho\sqrt{h''(\rho)}}\right)$, uniformly for $\pi > |\theta| > \epsilon$.

## 2 Known results

Considering a generating function $\phi$ with non negative real Taylor coefficients at the origin which tends to infinity "rapidly enough" at its finite or infinite singularity, it is not difficult to show that the saddle-point equation (1) has roots. Using the triangular inequality, one gets that one of the saddle-points of smallest modulus is a real positive number, and that its modulus is less than the radius of convergence of $\phi$. If, besides, the gcd of the indices of the non-zero coefficients of $\phi$ is 1, then this saddle-point is the only one with this modulus.

What makes it hard to get a majorisation of $\phi$ on the circle of radius $\rho$ (condition *(c.)* above) is that when $n$ tends to infinity, this circle can get close to singularities and other saddle-points of the function. This happens for instance with $\phi(z) = z + e^{z^2}$.

When $d$ is fixed, this problem is solved for large classes of functions (called "admissible") either by requiring the functions to satisfy more or less stringent conditions (see [6, 7, 9]), or by taking into account the other contributions to the integral [12].

The effect of $d$ tending to infinity when $\phi = f^d$ is *i)* to increase the difference between the largest value of $f$ on the contour and the other ones, *ii)* to make $\phi$ grow fast enough; both properties make it easier to prove the validity of the saddle-point method. Thus with the following hypotheses:

**H1:** $f$ has non negative real coefficients and $\gcd\{k \mid [z^k]f(z) \neq 0\} = 1$;

**H2:** $0 < a < d/n < b$ for some real $a$ and $b$ (depending on $f$);

Daniels has proved in [2] that the saddle-point method applies. This result is reformulated and slightly improved (without proof) in [5][5].
What makes hypothesis **H1** very useful is the following lemma.

**Lemma 1** *Let $f$ be a generating function with radius of convergence $R \leq \infty$. If $f$ satisfies* **H1**, *then for any $r < R$, $\theta \mapsto |f(re^{i\theta})|$ has a single maximum in $(-\pi, \pi)$, which is attained for $\theta = 0$.*

This was proved by Daniels and probably many others before.

In another context, Meir and Moon have proved in [8] the applicability of the saddle-point method to the coefficients of $f^d$ when $f$ is solution to $f(x) = x\phi(f(x))$, where $\phi$ satisfies **H1** and $\phi(0) = 1$, with the following restriction on $d$: $d = \alpha n + \lambda\sqrt{n} + o(n^{1/2})$, $\alpha$ being possibly zero. Intuitively, the reason for this is that the saddle-point method already applies when $d = 1$ (see [11]).

## 3 The case $n = o(d)$

This is the case studied by D. Gardy in this talk, following [3]. Then the smallest solution of (1) obviously tends to zero, and one can even give an asymptotic estimate for $\rho$. All the computations then get simpler and particularly the determination of a domain in which the function is "sufficiently" Gaussian. D. Gardy's main theorem is the following.

**Theorem 1** *Let $f$ satisfy* **H1** *and $n = o(d)$. Then (1) has a unique real positive root $\rho$, and asymptotically*

$$[z^n]f^d(z) = \frac{f^d(\rho)}{\rho^n\sqrt{2\pi n}}(1 + o(1)), \quad n \to \infty.$$

The proof strategy is as follows:

1. first prove the existence of $\rho$ by an intermediate value argument and compute an asymptotic estimate by inversion:
$$\rho = \frac{f(0)}{f'(0)}\frac{n}{d}(1 + O(n/d));$$
2. from this deduce an estimate for $f^{(k)}(\rho) = f^{(k)}(0) + O(n/d)$ and
$$\rho^2 h''(\rho) = n(1 + O(n/d) + O(1/n)), \quad \frac{\rho h^{(3)}(\rho e^{i\theta})}{h''(\rho)} = -2 + O(n/d) + O(\theta);$$
3. these estimates show that the function $\epsilon(n) = n^{-1+1/k}$ for any integer $k > 1$ satisfies conditions *(a.)* and *(b.)*, and a few extra computations with the help of Lemma 1 show that *(c.)* is also fulfilled.

D. Gardy then proceeds with the study of $[z^n]f^d(z)\psi(z)$ with the same hypothesis on $f$ and $d$ and a function $\psi$ analytic and non-zero at the origin, to be specified. In this case, it is known that the saddle-point method still works with the same saddle-point and the same function $\epsilon(n)$ as before provided that

*(d.)* $|\psi(\rho e^{i\theta})| \sim \psi(\rho)$ uniformly for $0 \leq |\theta| \leq \epsilon$ or;

[5] As it is stated, Good's theorem (p.868) is wrong, a counter-example being $[z^n](1 + z)^n$.

*(d'.)* uniformly for $0 \leq |\theta| \leq \epsilon$, the following estimates hold:

$$\left|\frac{\psi'}{\psi}\right|(\rho) = o(\sqrt{h''(\rho)}), \qquad \left|\left(\frac{\psi'}{\psi}\right)'\right|(\rho e^{i\theta}) = o(h''(\rho e^{i\theta}));$$

*(e.)* $|\psi(\rho e^{i\theta})| = O(\psi(\rho))$ uniformly for $\epsilon < |\theta| < \pi$;

and then the asymptotic estimate has to be multiplied by $\psi(\rho)$. Conditions *(d.)* and *(e.)* are obviously fulfilled if $\psi$ is an analytic function which does not depend on $d$ or $n$, but more generally, D. Gardy considers functions $\psi$ satisfying **H1** and of the form

$$\psi(z) = \prod_{i=1}^{p} g_i(z)^{d_i}, \quad d_i = o(d/\sqrt{n}).$$

Since $\psi(0) \neq 0$, it is easy to see that *(d'.)* is then satisfied, and *(e.)* follows immediately from Lemma 1.

**Note.** The sufficient conditions used in this summary are taken or derived from [1].

## References


[1] L. Berg. Asymptotics of integrals, series, and operators. In R. Wong, editor, *Asymptotic and Computational Analysis*, volume 124 of *Lecture notes in pure and applied mathematics*, pages 35–52. M. Dekker, New York, Basel, 1990. Conference in Honor of Frank W. J. Olver's 65th Birthday.

[2] H. E. Daniels. Saddlepoint approximations in statistics. *Annals of Mathematical Statistics*, 25:631–650, 1954.

[3] D. Gardy. On coefficients of powers of functions. Technical Report CS-91-53, Brown University, August 1991.

[4] D. Gardy. Some results on the asymptotic bahaviour of coefficients of large powers of functions. Rapport de recherche 769, Laboratoire de recherche en informatique, Université de Paris Sud, August 1992.

[5] I. J. Good. Saddle-point methods for the multinomial distribution. *Annals of Mathematical Statistics*, 28:861–881, 1957.

[6] B. Harris and L. Schoenfeld. Asymptotic expansions for the coefficients of analytic functions. *Illinois Journal of Mathematics*, 12:264–277, 1968.

[7] W. K. Hayman. Coefficient problems for univalent functions and related function classes. *Journal of the London Mathematical Society*, 40:385–406, 1965.

[8] A. Meir and J. W. Moon. The asymptotic behaviour of coefficients of powers of certain generating functions. *European Journal of Combinatorics*, 11:581–587, 1990.

[9] A. M. Odlyzko and L. B. Richmond. Asymptotic expansions for the coefficients of analytic generating functions. *Aequationes Mathematicae*, 28:50–63, 1985.

[10] L. Sirovich. *Techniques of Asymptotic Analysis*. Springer Verlag, 1971.

[11] J.-M. Steyaert and Ph. Flajolet. Patterns and pattern-matching in trees: an analysis. *Information and Control*, 58(1–3):19–58, July 1983.

[12] M. Wyman. The asymptotic behavior of the Laurent coefficients. *Canadian Journal of Mathematics*, 11:534–555, 1959.

# 16

# Transformée de Mellin et asymptotique : le tri-fusion

Mordecai Golin
INRIA, Rocquencourt

[résumé par Philippe Dumas]

Le tri-fusion est l'archétype des algorithmes du type "diviser pour régner" et sa complexité, mesurée en nombre de comparaisons, est depuis longtemps étudiée [4, 5]. Si $T(n)$ et $U(n)$ désignent respectivement la complexité dans le cas le pire et dans le cas moyen, l'utilisation des récurrences

$$T(n) = T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + n - 1,$$
$$U(n) = U(\lfloor n/2 \rfloor) + U(\lceil n/2 \rceil) + n - \gamma_n,$$

avec les conditions initiales $T(1) = U(1) = 0$, et $\gamma_n = \dfrac{\lfloor n/2 \rfloor}{\lceil n/2 \rceil + 1} + \dfrac{\lceil n/2 \rceil}{\lfloor n/2 \rfloor + 1}$, permet de montrer que

$$T(n) = n \lg n + nA(\lg n) + 1,$$

où $A(u)$ est la fonction 1-périodique $1 - \{u\} - 2^{1-\{u\}}$, et

$$U(n) = n \lg n + \beta n + O(1),$$

si $n = 2^k$, avec $\beta = -\displaystyle\sum_{j \geq 0} \frac{1}{2^j + 1} = -1{,}26449.$

La première formule est satisfaisante, mais il n'en est pas de même de la seconde qui ne concerne que les puissances de 2. De plus on se contente souvent de prouver que $U(n)$ est en $O(n \lg n)$ [1]. Le but de cet exposé est de montrer comment obtenir un développement asymptotique plus précis de $U(n)$, par des méthodes de théorie des nombres, qui sont applicables à beaucoup de récurrences de la forme "diviser pour régner".

**Formule de Perron.** L'outil de base est la formule de Perron [6, p. 151], dans une version adaptée au problème. Elle concerne la série de Dirichlet associée à une suite $(w_n)$,

$$W(s) = \sum_{n=1}^{\infty} \frac{w_n}{n^s}.$$

**Lemme 1** *Si la série de Dirichlet $W(s)$ converge absolument pour $\Re(s) > 2$, alors*

$$\frac{n}{2i\pi} \int_{3-i\infty}^{3+i\infty} W(s) n^s \, \frac{ds}{s(s+1)} = \sum_{k=1}^{n-1} (n-k) w_k.$$

Cette formule permet d'étudier des fonctions arithmétiques liées à l'écriture des entiers dans une base donnée suivant les méthodes exposées ici [2]. En l'appliquant à la suite $w_n = \Delta\nabla f_n$, où $\Delta$ et $\nabla$ sont respectivement les opérateurs de différence avant et arrière, on obtient le lemme de base qui va permettre d'étudier la complexité du tri-fusion.

**Lemme 2** *Si la suite $(f_n)$ vérifie la récurrence*

$$f_n = f_{\lfloor n/2 \rfloor} + f_{\lceil n/2 \rceil} + e_n,$$

*pour $n \geq 2$, avec $f_1$ donné et $e_n = O(n)$, alors $f_n$ admet l'expression intégrale*

$$f_n = nf_1 + \frac{n}{2i\pi} \int_{3-i\infty}^{3+i\infty} \frac{\Xi(s) n^s}{1-2^{-s}} \frac{ds}{s(s+1)},$$

*où*

$$\Xi(s) = \Delta\nabla f_1 + \sum_{n=2}^{\infty} \frac{\Delta\nabla e_n}{n^s}.$$

**Cas le pire.** Ce résultat permet de revenir sur la complexité dans le cas le pire. On a alors $f_n = T(n)$ et

$$\frac{f_n}{n} = \frac{1}{2i\pi} \int_{3-i\infty}^{3+i\infty} \frac{n^s}{1-2^{-s}} \frac{ds}{s(s+1)}.$$

L'intégrande est une fonction méromorphe dans le plan complexe dont les pôles sont $0$, $-1$ et les $\chi_k = 2ik\pi/\ln 2$ $(k \neq 0)$. Le premier est double alors que les autres sont simples. En poussant la droite d'intégration vers la gauche la collecte des résidus donne l'énoncé suivant. Dans ce cas particulier $f_n/n$ est exactement égal à la somme de tous les résidus.

**Théorème 1** *La complexité du tri-fusion dans le cas le pire, $T(n)$, satisfait*

$$T(n) = n \lg n + nA(\lg n) + 1$$

*où $A(u)$ est une fonction 1-périodique de valeur moyenne*

$$a_0 = \frac{1}{2} - \frac{1}{\log 2} \simeq -0,94269\,50408$$

*et $A(u)$ a un développement de Fourier explicite,*

$$A(u) = \sum_{k \in \mathbb{Z}} a_k e^{2ik\pi u},$$

*où pour $k \neq 0$*

$$a_k = \frac{1}{\log 2} \frac{1}{\chi_k(\chi_k+1)}, \quad \chi_k = \frac{2ik\pi}{\log 2}.$$

**Cas moyen.** Le cas moyen se traite de la même façon et donne un résultat non élémentaire, contrairement au précédent.

**Théorème 2** *Le coût moyen du tri-fusion, $U(n)$, est donné par*

$$U(n) = n \lg n + nB(\lg n) + O\left(n^{\epsilon}\right),$$

*pour tout $\epsilon > 0$. La fonction $B(u)$ est continue non dérivable et 1-périodique. De plus*

1. *la valeur moyenne de $B(u)$ est*

$$b_0 = \frac{1}{2} - \frac{1}{\log 2} - \frac{1}{\log 2} \sum_{m=1}^{\infty} \frac{2}{(m+1)(m+2)} \log\left(\frac{2m+1}{2m}\right) \simeq -1{,}24815;$$

2. *les coefficients de Fourier de $B(u)$ sont pour $k \neq 0$*

$$b_k = \frac{1}{\log 2} \frac{1 + \Psi(\chi_k)}{\chi_k(\chi_k + 1)}$$

*avec* $\chi_k = \dfrac{2ik\pi}{\log 2}$ *et* $\Psi(s) = \displaystyle\sum_{m=1}^{\infty} \frac{2}{(m+1)(m+2)} \left[\frac{-1}{(2m)^s} + \frac{1}{(2m+1)^s}\right]$ ;

3. *la série de Fourier est uniformément convergente vers $B(u)$ ;*
4. *les valeurs extrêmes de $B(u)$ sont $\beta \simeq -1,26449$ et $-1,24075 \pm 10^{-5}$.*

Ici on ne peut pousser la droite d'intégration que jusqu'à l'abscisse $-1 + \epsilon$, d'où la présence du terme $O(n^{\epsilon})$.
Les deux fonctions périodiques $A(u)$ et $B(u)$ sont reliées d'une jolie façon. En effet, en posant $A^*(u) = A(u) - a_0$ et $B^*(u) = B(u) - b_0$, on a

$$B^*(u) = A^*(u) + \sum_{m=1}^{\infty} \psi_m A^*(u - \lg m),$$

où les $\psi_m$ sont les coefficients de la série de Dirichlet $\Psi(s)$, $-\psi_{2m} = \psi_{2m+1} = \dfrac{2}{(m+1)(m+2)}$. La fonction $A^*(u)$ possède un point de rebroussement en $u = 0$, qui se reproduit dans $B^*(u)$ par la présence de $A^*(u)$ mais aussi des pseudo-harmoniques $A^*(u - \lg 2)$, $A^*(u - \lg 3)$, *etc*, ce qui donne à la partie périodique de $U(n)$ son aspect fractal que l'on peut constater sur la figure 1.

**Conclusion.** La formule de Perron permet d'étudier le comportement asymptotique des suites récurrentes du type "diviser pour régner". Cependant la série de Dirichlet n'est pas toujours explicite et le niveau de sommation à employer, de façon à assurer la convergence des intégrales, dépend de la suite.

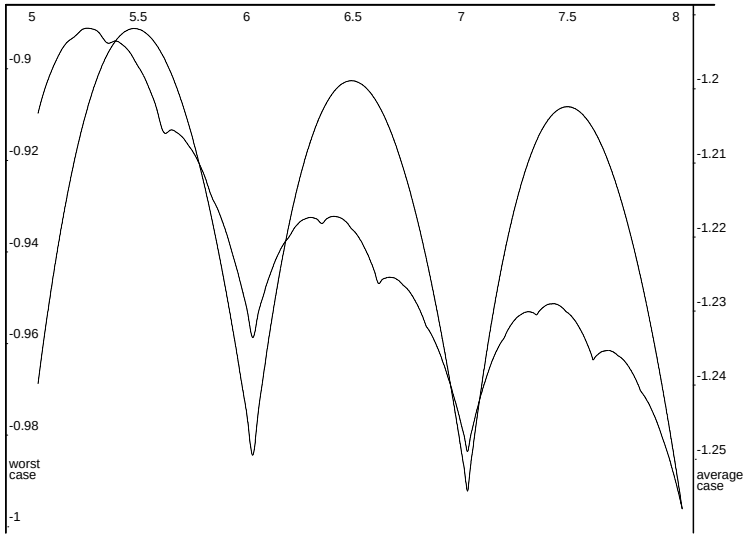



Figure 1 : Les complexités du tri-fusion dans le cas le pire et dans le cas moyen cachent des fonctions périodiques en $\lg n$, qui apparaissent quand on considère les suites $f_n/n - \lg n$ en fonction de $\lg n$. Le tracé le plus lisse correspond au cas le pire. Pour le cas moyen on a une courbe de nature fractale que l'on peut obtenir en superposant des copies réduites de la précédente décalées de $\lg 2, \lg 3$, *etc.*

## Références

[1] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms.* McGraw-Hill, New York, 1990.

[2] P. Flajolet, P. Grabner, P. Kirschenhofer, H. Prodinger, and R. Tichy. Mellin transforms and asymptotics: digital sums. Technical Report 1498, Institut National de Recherche en Informatique et en Automatique, 1991. To appear in *Theoretical Computer Science*, December 1993.

[3] Ph. Flajolet and M. Golin. Mellin transforms and asymptotics: The mergesort recurrence. Research Report 1612, Institut National de Recherche en Informatique et en Automatique, January 1992.

[4] D. E. Knuth. *The Art of Computer Programming*, volume 3 : Sorting and Searching. Addison-Wesley, 1973.

[5] R. Sedgewick. *Algorithms.* Addison-Wesley, Reading, Mass., second edition, 1988.

[6] G. Tenenbaum. *Introduction à la théorie analytique et probabiliste des nombres.* Institut Élie Cartan, Nancy, 1992.

# 17

# Asymptotique de récurrences et dénombrement de partitions

Philippe Dumas
INRIA, Rocquencourt

[résumé par François Lassner]

## 1 Position du problème général

Sous ce titre se cache une étude fine de relations de récurrence issues de méthodes "diviser pour régner" d'une part dont le prototype est $u_n = u_{n-1} + u_{\lfloor n/2 \rfloor}$ mais aussi de travaux déjà anciens de K. Mahler en théorie des nombres; il s'agissait d'étudier les fonctions analytiques $f(z) = \sum_{n\geq 0} a_n z^n$ telles qu'il existe des polynômes $c_k(z)$, $0 \leq k \leq N$, $N$ étant fixé, vérifiant

$$\sum_{k=0}^{N} c_k(z) f(z^{2^k}) = b(z). \tag{1}$$

Une identification donne $c_{00}a_n + c_{01}a_{n-1} + c_{02}a_{n-2} + \cdots + c_{10}a_{\lfloor \frac{n}{2} \rfloor} + c_{11}a_{\lfloor \frac{n-1}{2} \rfloor} + \cdots + \cdots = b_n$. Ce sont d'ailleurs les termes $c_{10}a_{\lfloor \frac{n}{2} \rfloor} + c_{11}a_{\lfloor \frac{n-1}{2} \rfloor} + \cdots$ qui rappellent la méthode "diviser pour régner". La question est l'évaluation asymptotique des $a_n$.

## 2 Un problème plus particulier et une classification

Philippe Dumas suggère de particulariser (1) en imposant d'abord $b(z) = c_2(z) = c_3(z) = \cdots = c_N(z) = 0$, $c_1(z) = -1$, $c_0(z) = \varphi(z)$ et que l'ensemble $Z(\varphi)$ des racines de $\varphi$ se place simplement par rapport au disque unité. La relation (1) se réduit alors à

$$\varphi(z) f(z) = f(z^2)$$

dont une solution formelle est $f(z) = \prod_{k\geq 0} \frac{1}{\varphi(z^{2^k})}$. Quelques exemples suggèrent une classification.
Cas 1 : les racines de $\varphi$ sont dans le disque unité. Si $\varphi(z) = 1 - 2z$ i.e. $Z(\varphi) = \{1/2\}$ alors $Z(\varphi(z^2)) = \{\pm 1/\sqrt{2}\}$ et les racines $> 0$ de $\varphi(z^{2^n})$ forment une suite de réels tendant vers 1 et $< 1$. Un calcul banal donne alors $a_n = 2^n + O(2^{n/2})$.
Cas 2 : les racines de $\varphi$ sont de module 1.

- $1^{\text{er}}$ cas particulier : $\varphi(z) = z + 1$ alors $f(z) = \prod_{k\geq 0} \frac{1}{1+z^{2^k}} = 1 - z$ car, d'après une remarque ancienne d'Euler, $\frac{1}{1-z} = \prod_{k\geq 0}(1 + z^{2^k})$ pour $|z| < 1$. Dans ce cas $a_n = 0$ pour $n \geq 2$.
- $2^{\text{ème}}$ cas particulier : les racines de $\varphi$ situées sur le cercle unité sont d'ordre pair au sens de la théorie des groupes, par exemple $\varphi(z) = 1 + z^2$ alors $f(z) = \prod_{k\geq 0} \frac{1}{1+z^{2^k}}$; dans ce cas on sait montrer qu'il existe un réel $c$ avec $a_n = O(n^c)$.

- $3^{\text{ème}}$ cas particulier : $\varphi(z) = 1 + z + z^2 = (z - j)(z - j^2)$ alors $f(z) = \prod_{k\geq 0} \frac{1}{1+z^{2^k}+z^{2.2^k}}$. La réunion $\bigcup_{k\geq 0} Z(\varphi(z^{2^k}))$ forme un sous-ensemble dense du cercle unité et $|z| = 1$ est une frontière naturelle de $f(z)$. L'étude des $a_n$ est exposée plus loin et la méthode s'applique plus généralement à $\varphi(z) = \Phi_a(z)$ avec $a$ impair.

Cas 3 : les racines de $\varphi$ sont à l'extérieur du cercle unité, par exemple $\varphi(z) = 1 - z/2$ alors $Z(\varphi) = \{2\}$ et $\bigcup Z(\varphi(z^{2^k})) = \cup_{k\geq 0}\{2^{1/2^k}\}$. On obtient un ensemble de points qui s'accumulent sur le cercle unité par l'extérieur. Ph. Dumas conjecture que $a_n$ est plus petit dans ce cas que dans celui qui est présenté ci-dessous.
On voit bien que le comportement de $f(z)$ et des $a_n$ est intimement lié à la géométrie des zéros de $\varphi$ i.e. à $Z(\varphi)$.

## 3 Étude détaillée de $f(z) = \prod_{k\geq 0} \frac{1}{1 + z^{2^k} + z^{2.2^k}} = \sum_{n\geq 0} a_n z^n$

On découvre aisément la nécessité de distinguer trois cas $n \equiv 0, 1, 2 \pmod 3$. L'étude devient beaucoup plus technique et utilise une variante de la méthode du col fondée sur la représentation $a_n = \frac{1}{2i\pi}\int_C \frac{f(z)}{z^{n+1}}dz$ où $C$ désigne un chemin fermé bien choisi, i.e. un cercle de rayon convenable $R = e^{-\rho}$. Ph. Dumas suggère plaisamment d'appeler la collecte approximative qu'il utilise la "méthode du faux-col". Un autre outil utilisé est la classique transformée de Mellin $M[\varphi(s)] = \int_0^\infty \varphi(t)t^{s-1}dt$ et en particulier $M[e^{-s}] = \Gamma(s)$. Citons aussi les séries de Dirichlet, la fonction de Möbius, les polynômes cyclotomiques. Par un mélange habile et complexe de ces outils, Ph. Dumas met en évidence un cercle de rayon $\rho$ tel que $f(z) = \sum_{n\geq 0} a_n z^n$ donne

$$a_n \sim \frac{1}{\sqrt{2\pi}} \exp\left[\frac{\ln^2 \rho}{2\ln 2} + (1 + K)\ln\rho + n\rho + \frac{1}{2}\ln\rho\right] \omega_0\left(\frac{\ln\rho}{2\ln 2}\right)$$

et le bon choix de $\rho$ est donné par $\frac{\ln\rho}{\ln 2} + n\rho + K = 0$ ce qui donne sans mal $\rho = -\frac{\ln n}{n\ln 2} + \frac{\ln\ln n}{n\ln 2} + O(1/n)$ ($\omega_0$ désigne une fonction 1-périodique).
Cette étude, très technique, suggère que ce sous-problème de (1) a déjà des solutions très variées et laisse peu d'espoir d'obtention de résultats globaux simples pour (1).
A la fin de l'exposé, Ph. Dumas montre l'existence de questions combinatoires connexes et des motivations informatiques liées, citant entre autres des travaux de De Bruijn; puis il signale des essais qu'il a menés sous Maple pour étudier empiriquement $a_n$.
Note de F. Lassner : Il me paraît utile de signaler une étude, assez complémentaire des travaux de Ph. Dumas, due à Erdős et assez ancienne (1941) portant sur $a_n = 1 + \sum_k a_{\lfloor n/\alpha_k \rfloor}$ et établissant que $a_n = \Theta(n^\rho)$ où $\rho$ vérifie $\sum_k \frac{1}{\alpha_k^\rho} = 1$; étude poursuivie en 1943 par l'étude de $P_r(n)$ nombre de partitions de $n$ en puissances de $r$ (obtenu aussi par Mahler en 1940) et plus près de nous la belle réponse au problème 1185 du journal Math. Mag. 58 (1984); la question étant l'étude de la suite $a_0 = 0$, $a_n = a_{\lfloor n/2 \rfloor} + a_{\lfloor n/3 \rfloor} + a_{\lfloor n/6 \rfloor}$ et la réponse, due à Erdős, Hildebrand, Odlyzko, Pudaite, Reznick est $\lim_{n\to\infty} \frac{a_n}{n} = \frac{12}{\ln 432}$. La méthode est une série de transformations menant de manière non banale à une équation de renouvellement c'est-à-dire à des méthodes taubériennes [théorèmes de Wiener et Ikehara].

## Références

[1] P. Erdős. On some asymptotic formulas in the theory of the "Factorisatio Numerorum". *Annals of Mathematics*, 42(4), October 1941.

[2] P. Erdős. Elementary proof of some asymptotic formulas in the theory of partitions. *Annals of Mathematics*, 43, 1942.

[3] P. Erdős. Corrections to two of my papers. *Annals of Mathematics*, 44(40), October 1943.

[4] P. Erdős and A. Odlyzko *et al.* Answer to problem 1185. *Math. Magazine*, 1985 and 1990.

[5] K. Mahler. On a special functional equation. *Journal of the London Mathematical Society*, 15, 1940.

[6] Rawsthorne. Problem 1185. *Math. Magazine*, page 42, 1984.

# 18

# Minorations de $\|(3/2)^k\|$

Laurent Habsieger
Université de Bordeaux I

[résumé par Luc Albert]

L'exposé consiste en une minoration de la distance de $(3/2)^k$ à l'entier le plus proche, notée $\|(3/2)^k\|$. L'origine de cette question est le problème de Waring c'est-à-dire trouver $g(k)$ minimal tel que

$$\forall n \in \mathbb{N}, \exists n_1, \ldots, n_{g(k)} \in \mathbb{N}, n = n_1^k + \cdots + n_{g(k)}^k.$$

En effet il a été prouvé que le $g(k)$ minimal est donné par la formule

$$g(k) = 2^k + \left[(3/2)^k\right] - 2$$

dès que

$$\left\{(3/2)^k\right\} \leq 1 - (3/4)^k \tag{1}$$

($[x]$ et $\{x\}$ dénotent respectivement la partie entière et la partie fractionnaire du réel $x$). Le problème revient donc à minorer $\|(3/2)^k\|$.
Historiquement, mis à part la minoration triviale $\|(3/2)^k\| \geq 1/2^k$ pour $k \geq 1$, on a attendu 1981 et Beukers pour montrer que $\|(3/2)^k\| \geq 1/2^{0.9k}$ pour $k \geq 5000$. Depuis, Dubitskas (1991) a obtenu $\|(3/2)^k\| \geq 0.5769^k \simeq 2^{-0.7936k}$ pour $k \geq k_0$ avec $k_0$ non explicite. Les résultats présentés ici sont les suivants.

**Théorème 1** *Pour $k$ assez grand (mais effectif) on a :*

$$\|(3/2)^k\| \geq 2^{-0.8471k}$$

*et pour $k \geq 5$, on a $\|(3/2)^k\| > 2^{-0.852k}$.*

Ces deux résultats, s'ils sont certes un progrès, restent encore éloignés du but idéal à atteindre qui serait d'obtenir l'inégalité (1) pour $k \geq 5$.
Les idées de preuve sont les suivantes: $3/2 = 1 + 1/2$, ou mieux $3/2 = 2 - 1/2$ et surtout $(3/2)^2 = 2 + 1/4$ (dans cette dernière expression, élevée à une puissance entière, de nombreuses simplifications vont en effet se produire).
La preuve du résultat utilise des approximants de Padé de $(1-t)^a$ en faisant varier le degré. Ceci met en jeu la fonction hypergéométrique de Gauss

$$F(\alpha, \beta, \gamma; t) = \sum_{n=0}^{\infty} \frac{(\alpha)_n (\beta)_n}{n!(\gamma)_n} t^n$$

(solution d'une équation différentielle du second ordre), la fonction $\Theta(x) = \sum_{p \leq x} \log p$ ($p$ premier) et le résultat $\Theta(x) \sim x$ au voisinage de l'infini. Les paramètres $\alpha$ et $\beta$ seront choisis afin d'optimiser le résultat.

La preuve du résultat comporte deux étapes avant l'obtention du théorème annoncé : une étude arithmétique et une étude asymptotique.
On commence par exprimer l'approximation de Padé pour la fonction $H(a,b,t)$ définie par

$$t^b H(a,b,t) = (1-t)^{a+b} - \sum_{n=0}^{b-1} \binom{a+b}{n} (-t)^n$$

avec $(a,b) \in \mathbb{N}^2$. L'approximation permet d'introduire les polynômes $Q_{p,q}, E_{p,q}, P_{p,q}$ tels que

$$P_{p,q}(a,b,t) - H(a,b,t)Q_{p,q}(a,b,t) = (-1)^{p+b} t^{p+q+1} E_{p,q}(a,b,t)$$

expression qui donne lieu à l'étude arithmétique. En plus de cette expression "binomiale" pour $Q$ et $E$, on a aussi l'expression intégrale de $Q_{p,q}(a,b,t)$ et $E_{p,q}(a,b,t)$ qui permet une étude asymptotique.
On considère le nombre premier $\pi$ tel que

$$\begin{aligned} \max(\frac{p+m+1}{2}, 2m-p, q+1) &\leq \pi \leq m + \frac{q}{3} \\ \text{ou } \max(p+m+1, 2m-p, q+1) &\leq \pi \leq \frac{3m+q}{2}. \end{aligned}$$

Ainsi on obtient que les polynômes $Q_{p,q}(2m,m,t)$, $P_{p,q}(2m,m,t)$ et $E_{p,q}(2m,m,t)$ appartiennent à l'idéal de $\mathbb{Z}[t]$ engendré par le produit $\Pi_m$ de ces tels $\pi$.
L'étude asymptotique nécessite des considérations de divisibilité, la connaissance du comportement de $\log \Pi_m$ et de majorations de $|Q_{p,q}(2m,m,-1/8)|$ et $|E_{p,q}(2m,m,-1/8)|$ obtenues à l'aide de leur expression intégrale.
La première partie du théorème est ainsi obtenue. Il est à noter que le choix optimal des paramètres $\alpha$ et $\beta$ est obtenu par essais en MAPLE.
La deuxième partie du théorème est obtenue pour des choix particuliers des paramètres $\alpha$ et $\beta$ qui permettent d'expliciter la borne du théorème précédent et d'affiner celle-ci en faisant une étude exhaustive pour les "quelques" cas restant (quand même de l'ordre de la centaine de mille).
Cette étude nous permet d'entrevoir le dur chemin à parcourir avant de clore le débat sur le problème de Waring.

## Références


[1] A. Baker and J. Coates. Fractional parts of powers of rationals. *Mathematical Proceedings of the Cambridge Philosophical Society*, 15:375–383, 1964.

[2] F. Beukers. Fractional parts of powers of rationals. *Mathematical Proceedings of the Cambridge Philosophical Society*, 90:13–20, 1981.

[3] F. Delmer and J.-M. Deshouillers. On the computation of $g(k)$ in Waring's problem. *Math. Comp.*, 54(190):885–893, 1990.

[4] A. Erdélyi. *Higher Trancendental Functions*, volume 1-2-3. R. E. Krieger publishing Company, Inc., Malabar, Florida, 2nd edition, 1981.

[5] J. Kubina and M. Wunderlich. Extending Waring's conjecture up to 471,600,000. *Math. Comp.*, 55(192):815–820, 1990.

[6] K. Mahler. On the fractional parts of powers of real numbers. *Mathematika*, 4:122–124, 1957.

[7] H. Rademacher. *Topics in Analytic Number Theory*. Springer-Verlag, 1973.

[8] J. B. Rosser and L. Schoenfeld. Approximate formulas for some functions of prime numbers. *Illinois Journal of Mathematics*, 6:64–94, 1962.

[9] J. B. Rosser and L. Schoenfeld. Sharper Bounds for the Chebyshev Functions $\theta(x)$ and $\psi(x)$. *Math. Comp.*, 29(129):243–269, 1975.

[10] L. Schoenfeld. Sharper Bounds for the Chebyshev Functions $\theta(x)$ and $\psi(x)$ II. *Math. Comp.*, 30(134):337–360, 1976.

# 19

# La recherche des racines complexes d'un polynôme selon Schönhage

Xavier Gourdon
École Polytechnique, Palaiseau

[résumé par Xavier Gourdon]

Le but de l'exposé est de présenter une méthode due à Schönhage qui recherche les racines complexes d'un polynôme à coefficients complexes [5, 6]. L'avantage de cette dernière est qu'elle permet *dans tous les cas* d'approcher les racines d'un polynôme. Ce n'est pas le cas avec les méthodes existantes. Des tests montrent par exemple que la méthode de Traub et Jenkins [3] qui est implantée en MAPLE et MATHEMATICA échoue lorsque le polynôme est mal conditionné (par exemple à racines multiples ou proches, ou à racines régulièrement réparties) ou lorsqu'il est de degré trop élevé (variant de 25 à 100 selon les polynômes et les programmes). (La méthode de Traub et Jenkins était jusque là reconnue comme étant la plus efficace). Une implantation soignée de l'algorithme de Schönhage en MAPLE donne sur tous les polynômes d'excellents résultats (pour une étude complète de l'algorithme, voir [2]).

## 1 Position du problème

**Notation 1** *On définit la norme* $|\cdot|$. *Si* $P = a_0 + a_1 X + \cdots + a_n X^n$, *on pose :*

$$|P| = |a_0| + |a_1| + \cdots + |a_n|.$$

Si tant d'algorithmes échouent, c'est que la recherche des racines d'un polynôme est un problème extrêmement mal conditionné. Le théorème de perturbation qui suit confirme ce fait.

**Théorème 1 (Ostrowski)** *Soit* $P$ *et* $Q$ *deux polynômes unitaires de degré* $n > 0$, *avec* $Q = (X - v_1)\cdots(X - v_n)$. *Alors si* $\epsilon = |P - Q|$, *on peut écrire* $P = (X - u_1)\cdots(X - u_n)$ *avec*

$$\forall i, \quad |u_i - v_i| < 4\rho\epsilon^{1/n}, \qquad \text{où } \rho = \sup\{1, |u_i|\}.$$

Autrement dit, les racines d'un polynôme bougent en $\epsilon^{1/n}$ lorsque ses coefficients bougent en $\epsilon$ (un exemple facile où le phénomène se produit est le cas de $P = X^n - \epsilon$ et $Q = X^n$). Ce résultat est en fait très pessimiste dans le cas général. De manière intuitive, lorsqu'un polynôme est perturbé à $\epsilon$ près, une racine $u$ d'ordre $p$ (i.e. $p$ racines du polynôme sont proches de $u$, les autres sont éloignées de $u$) se déplace avec une erreur de l'ordre de $\epsilon^{1/p}$.
Le théorème justifie le choix fait par Schönhage. Un polynôme $P$ unitaire de degré $n > 1$ et un entier $s > 0$ étant donnés, son algorithme détermine $n$ nombres complexes $u_1, \ldots, u_n$ tels que :

$$|P - (X - u_1)\cdots(X - u_n)| < 2^{-s} \tag{1}$$

en un temps d'exécution

$$O\left[(n^3 \log n + sn^2)\log(ns)\log(\log(ns))\right]. \tag{2}$$

C'est donc au sens de (1) que l'algorithme donne des approximations des racines de $P$. En généralisant le théorème donné plus haut, on peut alors donner *a posteriori* pour chaque racine trouvée une borne d'erreur proche de l'imprécision réelle. Quant à (2), ce résultat est surtout théorique. Il est obtenu en utilisant des méthodes de F.F.T. pour le calcul de produit et de division de polynômes, et la constante devant (2) est énorme. Dans la pratique, c'est surtout le fait que les racines soient trouvées dans *tous les cas* qui est intéressant.

## 2 Principe de la méthode de Schönhage

Soit $P$ un polynôme à coefficients complexes de degré $n > 1$. L'idée est non pas de calculer directement une racine de $P$ (comme le font en général les autres méthodes), mais de factoriser $P$ en deux polynômes : $P = FG$. En réitérant ainsi le procédé sur $F$ et $G$, on saura écrire $P$ comme produit de facteurs du premier degré.
L'algorithme détermine d'abord un cercle $\Gamma$ contenant une partie des racines de $P$ (voir figure 1), par exemple

$$u_1, u_2, \ldots, u_k \qquad \text{avec} \qquad 1 \le k < n.$$

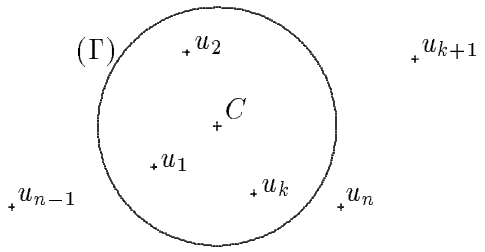



Figure 1. Les racines de $P$ et le cercle de séparation.

Par intégration le long de ce cercle (appelé cercle de séparation), on peut ensuite déterminer le polynôme unitaire dont les racines sont $u_1, u_2, \ldots, u_k$. On a en effet :

$$\forall p, \qquad S_p = u_1^p + u_2^p + \cdots + u_k^p = \oint_{(\Gamma)} \frac{P'(z)}{P(z)} z^p \, dz \tag{3}$$

d'où les coefficients de $F(z) = (z - u_1)\cdots(z - u_k)$ par les formules de Newton. Si $G$ est le quotient de la division euclidienne de $P$ par $F$, on a donc factorisé $P$ en un produit de deux facteurs. Ceci dit, l'approximation numérique de (3) par discrétisation de $\Gamma$ sera plus ou moins précise selon le choix de $\Gamma$. On sent bien que s'il se trouve près de $\Gamma$ une racine de $P$, le calcul numérique de (3) sera mauvais. Dans cette optique, le cercle de séparation $\Gamma$ sera choisi de sorte que les racines de $P$ s'en éloignent le plus possible.

# 3 Recherche du cercle de séparation

**Notation 2** *Si $P$ est un polynôme complexe de degré $n > 0$, on note $r_1(P), \cdots, r_n(P)$ les modules de ses racines rangés dans l'ordre croissant.*

On veut un cercle de séparation pour $P$, i.e. un cercle contenant une partie non vide de ses racines et ne les contenant pas toutes. Si $\Delta = \log(r_n(P)/r_1(P)) > 0$, alors $\exists j$ tel que $\log(r_{j+1}(P)/r_j(P)) > \Delta/(n-1)$. On a alors trouvé un candidat pour notre cercle de séparation : le cercle de centre 0 de rayon $\sqrt{r_j r_{j+1}}$. Ceci n'est malheureusement possible que si $\Delta \neq 0$; de toutes façons, même si $\Delta \neq 0$ il se peut que $\Delta$ soit faible et alors notre cercle de séparation n'est pas intéressant puisqu'il est trop près des racines. Nous allons pallier à ce problème en changeant d'origine.

## 3.1 Recherche du centre du cercle de séparation

On se ramène d'abord au cas où le barycentre des racines de $P$ est l'origine (pour cela, si $P = X^n + a_{n-1}X^{n-1} + \cdots$, considérer $P(X - a_{n-1}/n)$). On se ramène ensuite au cas où le plus grand des rayons des racines de $P$ est égal à 1 (voir le paragraphe suivant pour le calcul de $r_n(P)$). Le dessin suivant montre alors que si $v_0 = 2, v_1 = 2i, v_2 = -2, v_3 = -2i$, alors $\exists j$ tel que $\Delta_j = \log(r_n(P_j)/r_1(P_j)) > 0.30$ (où $P_j = P(X + v_j)$). En calculant $\Delta_j$ pour $j = 0 \ldots 3$, puis en choisissant $j$ maximisant $\Delta_j$, on s'est ainsi ramené au cas où $\Delta > 0.30$.

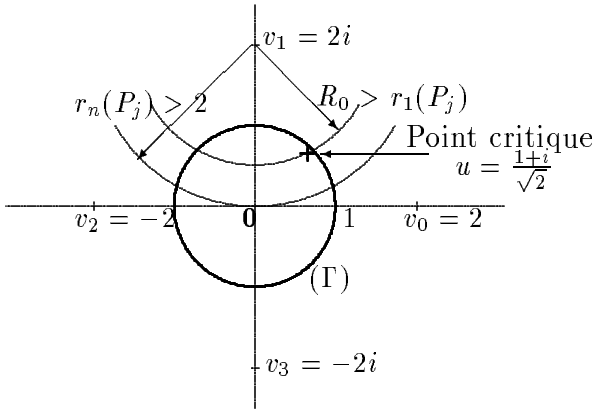



Figure 2. Les centres $v_i$ pour obtenir l'écartement $\Delta = \log(r_n/r_1) > 0.30$.

## 3.2 Recherche du rayon du cercle de séparation

Il nous faut savoir calculer les modules $r_i(P)$ des racines de $P$. Comme $\Delta = \log(r_n/r_1)$ n'est pas trop petit, on saura ensuite trouver un cercle de séparation dont les racines ne soient pas trop proches. Ceci est permis grâce à la méthode de Graeffe dont nous rappelons les grandes lignes.
L'idée est que les racines du polynôme $Q$ défini par $Q(X^2) = P(X)P(-X)$ sont les carrés des racines de $P$. On note $Q$ =Graeffe($P$). On pose $P_0 = P$ puis $P_m$ =Graeffe($P_{m-1}$), de sorte que $r_k(P_m) = r_k(P)^{2^m}$, et donc, si $r_1(P) < \cdots < r_n(P)$, en notant $P_m = a_n^{(m)}X^n + a_{n-1}^{(m)}X^{n-1} + \cdots$ :

$$\lim_{m\to\infty} \left| \frac{a_{k-1}^{(m)}}{a_k^{(m)}} \right|^{2^{-m}} = r_k(P).$$

L'utilisation directe de ce procédé est difficile pour deux raisons.

- La convergence n'est assurée que si $r_1 < \cdots < r_n$. Par ailleurs, même si cette dernière condition est réalisée, la convergence peut être très lente (c'est le cas si pour un $k$, $r_{k+1}/r_k$ est proche de 1).
- Les coefficients de $P_m$ divergent très rapidement en module.

Schönhage contourne ces difficultés en utilisant une approche modifiée du problème (pour plus de détails, voir [2, 3.1].

# 4 Calcul d'un facteur à partir du cercle de séparation

Quitte à effectuer des transformations simples, on peut supposer que le cercle de séparation est le cercle unité. Sa recherche nous permet même de connaître le nombre $k$ des racines se trouvant à l'intérieur du cercle et un paramètre $\delta > 0$ tel qu'aucune des racines de $P$ ne se trouve dans la couronne $\{e^{-\delta} < |z| < e^{\delta}\}$.
Les coefficients du polynôme $F$ dont les racines se trouvent à l'intérieur du cercle de séparation sont donnés par calcul de (3) suivi des formules de Newton. On approxime les valeurs $S_p$ de (3) pour $1 \leq p \leq k$ par discrétisation du cercle unité en $N$ points régulièrement répartis. On montre qu'après calculs, l'approximation $F_0$ de $F$ vérifie

$$|F_0 - F| < |F| kn e^{-\delta(N-k)}.$$

C'est cette majoration qui permet de déterminer la valeur de $N$ qu'il faut choisir pour être sûr que $F_0$ est déterminé avec suffisamment de précision. Seulement ce calcul peut devenir très coûteux si l'on désire avoir une bonne précision sur $F_0$. L'algorithme de Schönhage s'en sort comme suit. Il détermine d'abord par cette méthode une approximation $F_0$ de $F$ suffisamment bonne puis il lance une méthode dite de Newton-Schönhage qui permet d'obtenir rapidement à partir de $F_0$ une bonne approximation de $F$.

## 4.1 La méthode de Newton Schönhage

**Hypothèses.** $P = FG$ où les racines de $F$ sont à l'intérieur du cercle unité, celles de $G$ en dehors. On connaît une approximation $F_0$ de $F$, une approximation $G_0$ de $G$ (déterminée par division euclidienne de $P$ par $F_0$).
**Problème.** On veut trouver deux termes correcteurs $f$ et $g$ tels que $F_1 = F_0 + f$ et $G_1 = G_0 + g$ approchent $F$ et $G$ mieux que $F_0$ et $G_0$.

On a $F_1 G_1 = (F_0 + f)(G_0 + g) = F_0 G_0 + f G_0 + g F_0 + fg$. Le terme $fg$ étant du second ordre, si on choisit $f$ et $g$ tels que

$$P = F_0 G_0 + f G_0 + g F_0 \quad \Longleftrightarrow \quad \frac{P - F_0 G_0}{F_0 G_0} = \frac{f}{F_0} + \frac{g}{G_0}, \tag{4}$$

on aura $|P - F_1 G_1| = |fg| \leq |f| \cdot |g|$, donc du second ordre. $F_0$ et $G_0$ sont premiers entre eux et $f$ et $g$ dans (4) sont donc uniques. On peut les déterminer par exemple grâce à l'algorithme d'Euclide. Sans rentrer dans les détails, disons que le problème est que cette méthode est peu stable numériquement et Schönhage s'en sort autrement à partir d'une représentation intégrale de $f$, conséquence du théorème des résidus.

### 4.2 Description complète de la factorisation à partir du cercle de séparation

On se demande maintenant avec quelle précision on doit calculer $F_0$ pour assurer une convergence rapide de la méthode de Newton-Schönhage. On montre que le nombre de points $N$ pris sur le cercle unité pour approximer $F_0$ doit vérifier :

$$N \geq k + \frac{1}{\delta}\left(\log(100) + \log(k^3(n-k)^{3/2}n) + \frac{5}{2}\log(\gamma) + \log\left(\frac{1}{\mu}\right) + 3\log(|F|\cdot|G|)\right),$$

où $\gamma = \frac{1}{2\pi}\oint |dt/P(t)|$ et $\mu = \inf_{|z|=1}|P(z)|$. Le calcul de $\gamma$ et $\mu$ est réglé par approximation numérique. Il faut donner une majoration de $|F|\cdot|G|$. Schönhage utilise la majoration $|F|\cdot|G| < 2^n|P|$, donnée par Mignotte dans [4]. En fait, on peut donner une inégalité beaucoup plus fine :

$$|F|\cdot|G| < 2^{n/2}\sqrt{\binom{n}{k}}\,[P]_2, \quad \text{où} \quad [P]_2 = \left(\sum_{i=0}^{n}\frac{|a_i|^2}{\binom{n}{i}}\right)^{1/2}.$$

Cette inégalité est une conséquence d'un résultat démontré dans [1]. Cette étape d'intégration numérique est la plus coûteuse de l'algorithme de Schönhage. Son coût est déjà considérablement diminué en utilisant une F.F.T partielle. Il est également important de diminuer le plus possible la valeur de $N$. Celle donnée plus haut améliore déjà celle donnée par Schönhage lui-même dans [5]. On peut encore l'améliorer en augmentant la valeur de $\delta$, ce qui est rendu possible par une transformation homographique préalable : $z \mapsto (z-a)/(\bar{a}z-1)$.

## Conclusion

Les optimisations données à l'algorithme de Schönhage dans [2] permettent de gagner par rapport à l'algorithme initial un facteur temps de l'ordre de 10. Le programme écrit en MAPLE fait 1500 lignes. Il donne de bons résultats numériques sur *tous les polynômes* (même mal conditionnés), en un temps cependant plus élevé que les algorithmes déjà existants.
Les bornes d'erreur données *a posteriori* sur les approximations des racines sont garanties et autorisent l'introduction de moyens numériques dans le calcul formel. Cette idée est nouvelle et permettrait parfois de diminuer considérablement le coût de certaines opérations formelles.

## Références

[1] B. Beauzamy. Products of polynomials and a priori estimates for coefficients in poynomial decompositions. *Journal of Symbolic Computation*, 1992. To appear.

[2] X. Gourdon. Algorithmique du théorème fondamental de l'algèbre. Rapport de recherche, Institut National de Recherche en Informatique et en Automatique, 1992. À paraître.

[3] M. A. Jenkins and J. F. Traub. A three-stage variable-shift iteration for polynomial zeros and its relation to generalized Rayleigh iteration. *Numer. Math.*, 14:252–263, 1970.

[4] M. Mignotte. An inequality about factors of polynomials. *Math. Comp.*, 28(128), 1974.

[5] A. Schönhage. The fundamental theorem of algebra in terms of computational complexity. Technical report, Mathematisches Institut der Universität Tübingen, 1982.

[6] A. Schönhage. Equation solving in terms of computational complexity. In *Proceedings of the International Congress of Mathematicians*, pages 131–153, 1987. Berkeley, California, 1986.

# 20

# Algorithms for Computing Limits and Asymptotic Forms

John R. Shackell
University of Kent, Canterbury

[summary by Bruno Salvy]

Due to problems of cancellations, it is well known that computing the limit of a real function is as difficult as computing an asymptotic expansion. However, existing symbolic computation systems only perform some more or less generalised power series manipulation and are unable to compute difficult limits. The problem studied by J. Shackell in this talk is the computation of asymptotic forms for a real function of one variable which is given either explicitly by composition of the usual field operations, exponentials, logarithms (exp-log functions) or by integrals (liouvillian functions). There are two important steps to perform: *i) computing* the asymptotic expansion, *ii) proving* that this computation is valid for a well-defined class of functions.

## 1 Nested expansions

The following example helps understanding the type of difficulty one encounters when performing asymptotic expansions:

$$f(x) = \exp(x^{-1} + e^{-x\log x}) - \exp(x^{-1}), \quad x \to \infty.$$

The classical way to find the asymptotic expansion of such a function is to expand both terms and then add up the resulting expansions. The first subproblem one has to solve in order to do so is to prove that

$$\forall k > 0, \qquad e^{-x\log x} = o(x^{-k}), \quad x \to \infty.$$

The second problem is that if one only computes a finite number of terms of each expansion, the only result one will get is $f(x) = O(x^{-k})$, for all finite $k$. A better way to compute the expansion is to first determine that a cancellation will take place, then rewrite $f$ as

$$f(x) = \exp(x^{-1}) \cdot (\exp(x^{-x}) - 1),$$

and from there derive the expected result:

$$f(x) = \exp(x^{-1})[x^{-x} + \frac{x^{-2x}}{2!} + \cdots + O(x^{-kx})], \quad x \to \infty.$$

The difficulties we faced in this computation are those of the general case: comparing orders of growth, deciding whether a cancellation occurs and dealing with it. Except for the problem of deciding whether an elementary constant is zero or not (called the "constant problem"), these are all the difficulties of the general case. Note that the constant problem is central to symbolic computation. It has not been proved to be undecidable, but is certainly difficult since it is connected

to subtle conjectures in number theory, such as Schanuel's conjecture. However, in 1984 Dahn and Göring gave a non-constructive proof that the problem of computing a limit for an exp-log function could be reduced to the problem of comparing constants. Then in 1990, J. Shackell gave an actual algorithm [3] which shows that the procedure followed in the above example can be automated and will solve the general case. Our next section will discuss the proper setting for this and we shall concentrate on an intuitive description of the algorithm in the rest of this section.

The first thing to do is to determine in which asymptotic scale the expansion will take place. A prior step to this is to compute the set of the different orders of growth occurring in the expression of $f$.

A proper definition of order of growth, both practical and general is difficult to arrive at. The philosophy here is that sums and products are easy; the logarithm has a reducing role, and thus can only matter in the form of an iterated logarithm of the variable; all the problems have their source in the exponential. The proper attitude towards exponential is to avoid expanding it more than necessary, and this led J. Shackell in [2] to the definition of nested form and nested expansion. Using the classical notations $l_k(x)$ for the logarithm iterated $k$ times and likewise $e_k(x)$ for the iterated exponential, a *nested form* is a finite sequence $\{(s_i, \epsilon_i, m_i, d_i, \phi_i), i = 1 \ldots k\}$, where $s_i$ and $m_i$ are positive integers, $\epsilon_i$ is $\pm 1$, $d_i$ is a real number, and $\phi_i$ will be made more precise later. Such a sequence represents

$$e_{s_1}^{\epsilon_1}(l_{m_1}^{d_1}(x)\phi_1(x)),$$

and recursively

$$\phi_{i-1}(x) = e_{s_i}^{\epsilon_i}(l_{m_i}^{d_i}(x)\phi_i(x)),$$

with the additional constraint that $\phi_i$ is of a smaller order of growth than $l_{m_i}$, and that $\phi_k$ tends to a non-zero finite limit, plus some conditions ensuring that this cannot be reduced by simplifying an $\exp(\log(.))$. Another condition is that the $\phi_i$'s should belong to a Hardy field and we shall come back to this in our next section. Having thus defined a nested form, one defines a *nested expansion* as a sequence of nested forms $N_i$, such that $N_{i+1}$ is a nested expansion for $\phi_{i,k_i}$ minus its limit. These nested expansions should be viewed as a very general asymptotic scale.

An important point is that it is possible to compute effectively with these nested expansions. The ordering, the exponential and the logarithm are easy, the integral can be done but there are difficulties with constants of integration, and the product can be done provided one can compute addition, which is now the difficult operation, because of possible cancellations. To compute the asymptotic expansion of a sum, one first builds up the set of growth orders of the subexpressions of the sum, order it, and then starting from the largest order of growth $t_1$, compute the *shadow* of the expression with respect to it. This shadow is obtained in many cases (integrals are more complicated) by substituting zero for all the occurrences of $t_i$ with $i \geq 1$ in the expression. Now we appeal to an oracle to decide whether the resulting expression is identically zero or not. If it is, we proceed with the next growth order. The first $t_i$ for which the expression is not zero gives the scale in which the expansion must be performed. This argument is turned into an algorithm for all the exp-log functions in [3].

## 2 Hardy fields

Once a sketch of the algorithm is clear, it is necessary to determine a class of expressions for which it works, and to prove that it actually works. A good framework for doing this is provided by *Hardy*

*fields.*
Let $\mathcal{X}$ be the ring of germs at $\infty$ of $C^\infty$ functions. (Think of it as the set of possible asymptotic behaviours.) A *Hardy field* is a subring of $\mathcal{X}$ which is a field closed under differentiation.
The main constraint here is that non-zero elements of Hardy fields have to be invertible, and thus cannot have arbitrarily large zeros. Consequently, since their derivative belongs to the field, they have to be ultimately monotonic and tend to a possibly infinite limit. Also, differences of two (germs of) functions of a Hardy field are also in the field and possess a limit, so that this field is ordered. A short introduction to Hardy fields is given in [4]. If $f$ and $g$ are two elements of a Hardy field tending to infinity, they are said to be *comparable* when there exists a positive integer $n$ such that

$$f < g^n \quad \text{and} \quad g < f^n,$$

where the order is that of the field. Extending this by saying that $\pm f$ and $f^{-1}$ are comparable and that two elements tending to a non-zero finite limit are comparable yields a decomposition of the Hardy field into equivalence classes called *comparability classes* and denoted $\gamma(f)$. One should think of these classes as basic functions of an asymptotic scale. Their number minus one is called *the rank* of the field.
As an important special case of Hardy fields, *Rosenlicht fields* have been considered by J. Shackell. These are Hardy fields of finite rank, closed by $f \mapsto f^c$, for all real $c$. In [4] it is proved that *any element of a Rosenlicht field has a nested expansion.* Besides, an algorithm is given in [4] to compute a nested expansion of solutions of algebraic differential equations that lie in a Hardy field. Basically, from the order of the equation, one deduces all the possible asymptotic forms of its solutions lying in a Hardy field, and then one tries to substitute in the equation and either get a contradiction or an induction to a "simpler" problem.
A complementary algorithm, described in [2] will compute a nested expansion for liouvillian functions. This needs a little more algebra. If $\mathcal{F}$ is a Hardy field, we need a computable set $\mathcal{I}(\mathcal{F}) = \{t_1, \ldots t_n\}$ of representatives of its comparability classes. Recall that such a set is computable for a Hardy field containing the solutions of an algebraic differential equation. Now the algebraic analogue of *shadows* together with its complementary notion of *ghost* is defined by associating to an element $t \in \mathcal{F}$ tending to zero the set

$$\mathcal{I}_t(\mathcal{F}) = \{f \in \mathcal{F} \,;\, \exists \delta > 0, |f| < t^\delta\}.$$

Then a subfield $\mathcal{S}_t$ of $\mathcal{F}$ has the *shadow property with respect to* $t$ if its intersection with $\mathcal{I}_t(\mathcal{F})$ is $\{0\}$ and it is closed under relative differentiation ($a, b \in \mathcal{S} \Rightarrow a'/b' \in \mathcal{S}$). The $i$th shadow of $f \in \mathcal{F}$, $\eta_i(f)$ should be thought of as the part of $f$ which is in some $\mathcal{S}_{t_i}(\mathcal{F})$, and the $i$th-ghost is simply $f - \eta_i(f)$. Given a computable Hardy field $\mathcal{F}$, the extension by a function $\theta$ is computable provided that

*(i)* we can compute a $\mathcal{I}(\mathcal{F})$ monomial $T$ such that $\theta/T$ has a non-zero finite limit;

*(ii)* we can compute the $i$th shadows and ghosts $\phi_i$ and $\psi_i$ for $\theta/T$;

*(iii)* $\mathcal{S}_i(\mathcal{F})(\phi_1, \ldots, \phi_i)$ has the shadow property with respect to $t_i$ and $\psi_i \in \mathcal{I}_{t_i}$;

*(iv)* there is a zero-equivalence algorithm for $\mathcal{F}(\phi_1, \ldots, \phi_n, \theta)$.

We now display three important cases when this is possible (except perhaps *(iv)*):

**Exponential extensions** If $g \in \mathcal{F}$ then setting $\eta_i(\exp(g)) = \exp(\eta_i(g))$ works.

**Integral extensions** If $f \in \mathcal{F}$, then by a theorem of Hardy one can find $T$. The computation of the shadow depends on the relative orders of growth of $T$ and $t_i$ and can be done (see [2]).

**Algebraic extensions** If $P$ is a polynomial over $\mathcal{F}$ then it is known that the possible comparability classes of its solutions are those of $\mathcal{F}$, so one can just substitute an arbitrary monomial and compute indeterminate coefficients. This is the basis of the algorithm, together with a general form of Sturm's theorem.

## References

[1] G. H. Hardy. *Orders of Infinity*, volume 12 of *Cambridge Tracts in Mathematics*. Cambridge University Press, 1910.

[2] J. R. Shackell. Limits of Liouvillian functions. Preprint.

[3] J. R. Shackell. Growth estimates for exp-log functions. *Journal of Symbolic Computation*, 10:611–632, December 1990.

[4] J. R. Shackell. Rosenlicht fields. To appear in *Transactions of the American Mathematical Society*, 1991.

# Part IV

# Analysis of Algorithms and Data Structures

# 21

# Variétés d'arbres croissants

Bruno Salvy
INRIA, Rocquencourt

[résumé par Michèle Soria]

On appelle arbre croissant un arbre étiqueté dont les étiquettes croissent le long des branches. Ces arbres ont été utilisés comme représentations de permutations, comme structures de données informatiques et comme modèles probabilistes dans diverses applications.
Nous présentons une approche générale permettant le calcul de paramètres de ces arbres. Les fonctions génératrices de ces paramètres sont reliées à une équation différentielle ordinaire simple, qui est non-linéaire et autonome. Les méthodes d'analyse de singularité permettent alors d'analyser asymptotiquement des paramètres comme le degré de la racine, le nombre de feuilles, la longueur de cheminement et les hauteurs des nœuds avec des hypothèses très faibles sur la famille d'arbres étudiée.

## 1 Introduction

Un *arbre étiqueté* de taille $n$ est un arbre enraciné formé de $n$ nœuds qui sont étiquetés par des entiers distincts de $\{1, \ldots, n\}$. On appelle *arbre croissant* un arbre étiqueté dont les étiquettes croissent le long des branches.
On étudie ici l'asymptotique de caractéristiques de ces arbres, via leur fonctions génératrices. Cette étude est à rapprocher de celle des "familles simples d'arbres" de Meir et Moon. Cependant, à cause de la contrainte de croissance des étiquettes, les équations définissant les fonctions génératrices ne sont plus des équations algébriques simples, mais des équations différentielles algébriques.
Pour un grand nombre de paramètres, les fonctions génératrices sont reliées à une équation différentielle ordinaire simple, qui est non-linéaire et autonome :

$$Y'(z) = \phi(Y(z)) \ .$$

La combinaison de méthodes algébriques et analytiques permet d'analyser dans un cadre uniforme un grand nombre de paramètres comme le degré de la racine, le nombre de feuilles, la longueur de cheminement et les hauteurs des nœuds avec des hypothèses très faibles sur la famille d'arbres étudiée.
Les résultats résumés ici font l'objet d'un article de François Bergeron, Philippe Flajolet et Bruno Salvy [1].

## 2 Énumération d'arbres croissants

Étant donné un multi-ensemble d'entiers $\Omega$, contenant au moins 0 et un entier $\geq 2$, on appelle *variété d'arbres croissants* planaires (ou non planaires) sur $\Omega$ l'ensemble de tous les arbres croissants planaires (ou non planaires) dont les nœuds ont un degré extérieur dans $\Omega$.

La *fonction de degré* d'une variété d'arbres croissants planaires sur $\Omega$ est $\phi(u) = \sum_{d\in\Omega} u^d$; et pour les arbres non planaires $\phi(u) = \sum_{d\in\Omega} \frac{u^d}{d!}$.
Étant donnée $\mathcal{Y}$ une variété d'arbres croissants, on note $Y(z) = \sum Y_n \frac{z^n}{n!}$ sa série génératrice exponentielle, où $Y_n$ est le nombre d'arbres croissants de taille $n$.
Former une forêt planaire de $k$ arbres correspond à la série $Y^k(z)$ (et $\frac{1}{k!}Y^k(z)$ pour les forêts non planaires), et rajouter une racine avec étiquette minimale à une forêt de série $W(z)$ correspond à la série $\int_0^z W(z)$. On a donc le théorème suivant.

**Théorème 1** *La série génératrice exponentielle $Y(z)$ d'une variété d'arbres définie par la fonction de degré $\phi$ est définie implicitement par*

$$Y'(z) = \phi(Y(z)),\ Y(0) = 0, \qquad \text{soit} \qquad \int_0^{Y(z)} \frac{du}{\phi(u)} = z.$$

Exemples. Lorsque $\int \frac{du}{\phi(u)}$ et son inverse s'expriment en termes de fonctions spéciales, $Y(z)$ peut avoir une forme explicite. Par exemple les arbres strictement binaires croissants (permutations alternées) ont pour fonction de degré $\phi(u) = 1+u^2$, d'où $y(z) = \tan(z)$. Les arbres croissants $d$–aires ont pour fonction de degré $\phi(u) = (1+u)^d$, d'où $y(z) = -1+[1-(d-1)z]^{-1/(d-1)}$. Les arbres récursifs (arbres planaires croissants sans contrainte d'arité) ont pour fonction de degré $\phi(u) = \exp(u)$ d'où $y(z) = \log\frac{1}{1-z}$. Pour les arbres récursifs non planaires la fonction de degré est $\phi(u) = \frac{1}{1-u}$, d'où $y(z) = 1-\sqrt{1-2z}$. □

**Étude asymptotique.** Lorsque $Y(z)$ n'a pas de forme exacte, on a recours à l'analyse asymptotique pour obtenir un équivalent de $Y_n$ quand $n$ tend vers l'infini. La méthode consiste à trouver la (les) singularité(s) de module minimal de $Y(z)$, développer $Y(z)$ au voisinage de cette (ces) singularité(s), et enfin traduire ce développement sur les coefficients.

Le rayon de convergence de $Y(z)$ définie par le théorème 1 est

$$\rho = \int_0^{\infty} \frac{du}{\phi(u)}\ .$$

Lorsque $\phi$ est non périodique (i.e. il n'existe pas de série $\psi$ telle que $\phi(u) = \psi(u^p)$, pour $p \geq 2$), $\rho$ est l'unique singularité dominante (i.e. de module minimal) de $Y(z)$. Lorsque $\phi$ est de période $p \geq 2$, $Y(z) = zY^*(z^p)$, et la série $Y^*(z)$ a une unique singularité dominante en $\rho^{1/p}$.
Lorsque $\phi$ est un polynôme de degré $d \geq 2$ (la variété $\mathcal{Y}$ est alors dite *polynomiale*), en développant $1/\phi(u)$ au voisinage de l'infini, puis en intégrant, on trouve pour $Y \to \infty$

$$\int_Y^{\infty} \frac{du}{\phi(u)} \ = \ C.Y^{d-1}\left(1 + O(\frac{1}{Y})\right)\ .$$

Or $\int_Y^{\infty} \frac{du}{\phi(u)} = \rho - z$, et en inversant cette relation, on obtient le développement singulier de $Y(z)$ en fonction de $(\rho - z)^{1/(d-1)}$ :

$$Y(z) = \lambda(1 - z/\rho)^{-1/(d-1)} + o((1 - z/\rho)^{-1/(d-1)})\ .$$

L'analyse de singularité permet de traduire l'asymptotique des fonctions sur leurs coefficients :

$$\begin{array}{rclcrcl} Y(z) & = & (1 - z/\rho)^{\alpha} & \Rightarrow & [z^n]Y(z) & = & \frac{1}{\Gamma(-\alpha)}\rho^{-n}n^{-\alpha-1}\left(1 + O(\frac{1}{n})\right) \\ Y(z) & = & O((1 - z/\rho)^{\alpha}) & \Rightarrow & [z^n]Y(z) & = & O(\rho^{-n}n^{-\alpha-1}) \end{array}$$

Et l'on a donc finalement

**Théorème 2** *Le nombre d'arbres croissants de taille $n$ dans une variété $\mathcal{Y}$ dont la fonction de degré est un polynôme de degré $d$ vaut asymptotiquement*

$$\frac{Y_n}{n!} \sim \lambda \rho^{-n} n^{-(d-2)/(d-1)} .$$

Exemple. Les arbres ternaires croissants stricts ont pour fonction de degré $\phi(u) = 1 + u^3$. Il n'existe pas de solution explicite pour $Y(z)$, mais on détermine la singularité $\rho = 2\pi\sqrt{3}/9$, et $\frac{Y_n}{n!} \sim \frac{1}{\sqrt{2\pi}} \rho^{-n-1/2} n^{-1/2}$.

## 3 Distribution de paramètres

Les méthodes précédentes s'appliquent aussi à l'étude de différents paramètres sur les variétés d'arbres. Un paramètre est *inductif* s'il est défini par une relation

$$s[t] = f_{|t|} + \sum_{\tau \propto t} s[\tau] ,$$

pour une suite $(f_n)$, et où la somme est prise sur tous les sous-arbres $\tau$ à la racine de l'arbre $t$. Citons comme exemples de paramètres inductifs la taille ($f_n = 1$), le nombre de feuilles ($f_n = \delta_{n,1}$) et la longueur de cheminement ($f_n = n$).
Étant donnés un paramètre inductif $s$ et une variété $\mathcal{Y}$, la série génératrice exponentielle de $s$ est $S(z) = \sum_{t \in \mathcal{Y}} s[t] \frac{z^{|t|}}{|t|!}$.
Introduisant la relation de définition de $s$ dans la série double $Y_s(z,u) = \sum_{t \in \mathcal{Y}} u^{s[t]} z^{|t|}$, on obtient

$$S(z) = F(z) + \int_0^z S(t) \phi'(Y(t)) dt,$$

où la série $F(z)$ est un produit de Hadamard : $F(z) = \sum f_n Y_n z^n/n!$. La résolution de l'équation différentielle induite donne le résultat suivant.

**Théorème 3** *La série génératrice exponentielle d'un paramètre inductif $s$ défini par $s[t] = f_{|t|} + \sum_{\tau \propto t} s[\tau]$, sur une variété $\mathcal{Y}$ est*

$$S(z) = Y'(z) \int_0^z \frac{F'(t)}{Y'(t)} dt, \tag{1}$$

*où $F(z)$ est définie à partir de $(f_n)$ et $\mathcal{Y}$ par $F(z) = \sum f_n Y_n \frac{z^n}{n!}$.*

**Étude asymptotique.** L'équation (1) agit comme un "transformateur de singularité". Pour les paramètres inductifs *élémentaires*, i.e. tels que $f_n = C n^\alpha \log^r n$, la singularité de $S$ est aussi en $\rho$, et l'analyse asymptotique résulte de celle de $Y(z)$ combinée à l'analyse de singularité sur les produits d'Hadamard (voir article [1]).

**Théorème 4** *Soit $s$ un paramètre inductif sur la variété polynomiale $\mathcal{Y}$, tel que $f_n = C n^\alpha \log^r n$. La valeur moyenne de $s$ sur les éléments de taille $n$ de $\mathcal{Y}$ vaut asymptotiquement*

$$\overline{S_n} \sim \lambda n \psi(n) \qquad \text{avec} \qquad \psi(n) = \begin{cases} 1 & \text{si } \alpha < 1, \\ \log^{r+1} n & \text{si } \alpha = 1, \\ n^{\alpha-1} \log^r n & \text{si } \alpha > 1. \end{cases}$$

EXEMPLE. La longueur de cheminement est un paramètre inductif, avec $f_n = n$ donc $\alpha = 1$ et $r = 0$ ; d'où pour toutes les familles d'arbres croissants $\overline{S_n} \sim \lambda n \log n$ . En particulier $\lambda = 2$ pour les arbres binaires (Quicksort) et strictement binaires.

La même méthode s'applique à des paramètres qui ne sont pas exactement inductifs élémentaires. Par exemple le nombre moyen de nœuds d'arité $i$ dans un arbre de taille $n$ d'une variété $\mathcal{Y}$ vaut :

$$\overline{S_n^{(i)}} \sim \lambda_i n \quad \text{avec} \quad \lambda_i = \frac{\phi}{\rho} \int_0^\rho \frac{Y^i(t)}{Y'(t)} dt.$$

Et la probabilité que le degré de la racine soit $j$ dans un arbre de taille $n$ d'une variété polynomiale (de degré $d$) $\mathcal{Y}$ est

$$\pi_{nj} \sim \frac{\alpha_j}{n^{(d-j)/(d-1)}}.$$

De plus une extension de la méthode permet de trouver des distributions limites de paramètres. Par exemple, la profondeur d'un nœud aléatoire : asymptotiquement gaussienne avec moyenne et variance en $\log n$ :

$$\mu_n = \frac{d}{d-1} \log n + O(1), \quad \sigma_n^2 = \frac{d}{d-1} \log n + O(1).$$

## 4 Extensions

Les résultats présentés peuvent être étendus dans diverses directions. Il est possible de considérer des schémas algébriques plus larges, par exemple

$$Y' = H(z, Y(z)) \quad \mapsto \quad S(z) = U(z) \int_0^z \frac{F'(t)}{U(t)} dt.$$

Les variétés non polynomiales peuvent aussi être traitées dans ce cadre. Lorsque la fonction de degré, à coefficients positifs, devient singulière en $\sigma > 0$ ($\phi$ entière, ou avec des singularités à distance finie), le théorème 2 se généralise : le comportement singulier de $Y(z)$ est obtenu en inversant le développement de $\sum du/\phi(u)$ autour de sa singularité dominante $\rho = \int_0^\infty du/\phi(u)$. Et la forme asymptotique des coefficients de $Y(z)$ se déduit par analyse de singularité. On peut ainsi par exemple traiter les arbres récursifs planaires et non planaires.
L'approche développée sur des expressions intégrales agissant comme des "transformateurs de singularité" est très générale, et permet de traiter de manière unifiée un grand nombre de problèmes statistiques.

## Références

[1] F. Bergeron, Ph. Flajolet, and B. Salvy. Varieties of increasing trees. In J.-C. Raoult, editor, *CAAP'92*, volume 581 of *Lecture Notes in Computer Science*, pages 24–48, 1992. Proceedings of the 17th Colloquium on Trees in Algebra and Programming, Rennes, France, February 1992.

# 22

# Limit Distributions in Quadtrees

Thomas Lafforgue
LRI, Orsay

[résumé par Philippe Flajolet]

Quadtrees constitute a classical data structure for storing and accessing multidimensional data. It is proved here that, in all dimensions, the cost of a random search in a randomly grown quadtree has logarithmic mean and variance and that it is asymptotically distributed as a normal variate. The limit distribution property extends to quadtrees a result only known so far to hold for binary search trees.
The analysis is based on a technique of singularity perturbation analysis applied to linear differential equations satisfied by intervening bivariate generating functions.
The work described is based on a joint paper of Lafforgue and Flajolet [4].

## 1 Introduction

Quadtrees are a well known data structure discovered by Finkel and Bentley which is discussed in classical treatises on algorithms [5, 11] and examined in great detail in Samet's reference books [9, 10]. Their analysis has made visible progress over recent years [2, 3, 6, 7, 8].
The probabilistic model considered takes all data uniformly from the $d$–dimensional hypercube $\mathcal{Q} = [0,1]^d$. The search cost $D_n$ is defined as the cost—measured in internal nodes traversed—of searching a random point in a randomly grown quadtree of size $n-1$; it is also called the insertion depth of the $n$th node and is a random variable defined on the space $\mathcal{Q}^{n-1} \times \mathcal{Q} \cong \mathcal{Q}^n$.
The main result reported is that *$D_n$ converges in distribution to a Gaussian variate* when the size $n-1$ of the tree structure becomes large. Figure 1 illustrates the clear occurrence of this phenomenon already for low values of $n$.

**Theorem 1** *(i). The mean $\mu_n$ and standard deviation $\sigma_n$ of the cost $D_n$ of a random search in random quadtree of size $n-1$ in dimension $d \geq 1$ satisfy asymptotically*

$$\mu_n \sim \frac{2}{d} \log n \qquad \textit{and} \qquad \sigma_n \sim \sqrt{\frac{2}{d^2} \log n}. \tag{1}$$

*(ii). The distribution of $D_n$ converges in distribution to a normal variate: for all real $\alpha, \beta$,*

$$\Pr\{\alpha \leq \frac{D_n - \mu_n}{\sigma_n} \leq \beta\} \;\to\; \frac{1}{\sqrt{2\pi}} \int_\alpha^\beta e^{-x^2/2}\, dx \qquad (n \to \infty). \tag{2}$$

The mean was determined by Devroye and Laforest [2] and independently by Flajolet, Gonnet, Puech, and Robson [3].

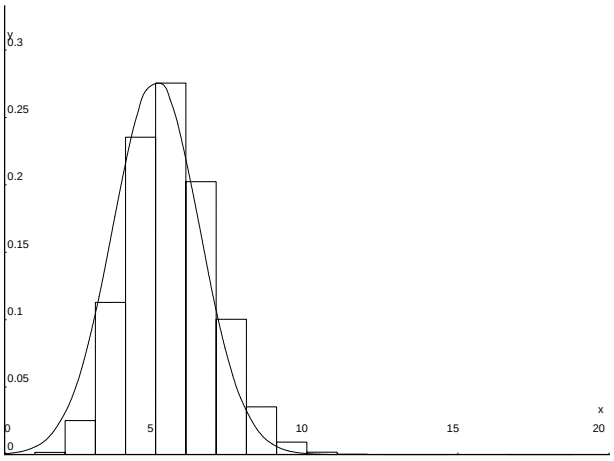



Figure 1: The histogram of the probability distribution of $D_n$ (for dimension $d = 2$ and $n = 100$) plotted against a Gaussian density function of same mean and variance.

## 2 Basic equations

Two integral operators play an essential rôle here:

$$\mathbf{I}\, f(z) = \int_0^z f(t)\, \frac{dt}{1-t} \qquad \mathbf{J}\, f(z) = \int_0^z f(t)\, \frac{dt}{t(1-t)}.$$

**Lemma 1** *The generating functions of the costs of a random search, successful ($C_n$) and unsuccessful ($D_n$), in a quadtree of size $n$ are given by*

$$\begin{cases} \gamma_n(u) & := & \displaystyle\sum_k \Pr\{C_n = k\} u^k = \frac{1}{n}\, \frac{u}{2^d u - 1}(\phi_n(u) - 1) \\ \delta_n(u) & := & \displaystyle\sum_k \Pr\{D_n = k\} u^k = \frac{1}{2^d u - 1}(\phi_n(u) - \phi_{n-1}(u)), \end{cases} \tag{3}$$

*where the bivariate generating function*

$$\Phi(u, z) = \sum_n \phi_n(u) z^n$$

*of the level polynomials $\phi_n(u)$ is characterized by the integral equation*

$$\Phi(u, z) = 1 + 2^d u \mathbf{J}^{d-1} \mathbf{I}\, \Phi(u, z). \tag{4}$$

**Proof.** A quadtree of size $n$ gives rise to a root subtree of size $k$ with probability

$$\pi_{n,k} = \frac{1}{n} \sum_{\mathcal{L}} \frac{1}{(\ell_1 + 1)(\ell_2 + 1) \cdots (\ell_{d-1} + 1)}, \tag{5}$$

where the summation is over $n > \ell_1 \geq \ell_2 \geq \cdots \geq \ell_{d-1} \geq k$. The rest relies on simple combinatorial properties of the "level polynomials" $\phi_n$ and on translating recurrences into generating function equations. □

## 3 Lower dimensions

*The binary search tree* ($d = 1$). The main results are originally due to Hibbard for the mean and Lynch for the whole distribution. See [8]. Related results hold for successful searches. When $d = 1$, the integral equation satisfied by $\Phi(u, z)$ is homogeneous of order 1, and thus solvable by quadratures. We find

$$\Phi(u, z) = \frac{1}{(1-z)^{2u}} \qquad \text{and} \qquad \phi_n(u) = \frac{(2u) \cdot (2u+1) \cdots (2u+n-1)}{n!}. \tag{6}$$

Thus, we have $[u^k]\phi_n(u) = 2^k \left[{n \atop k}\right]/n!$, which involves the Stirling numbers of the first kind ("cycle" Stirling numbers).

**Theorem 2 (Hibbard, Lynch)** *The cost $D_n$ of a random search in a binary search tree of size $n-1$ has mean and variance given by*

$$\mu_n = 2(H_n - 1) \qquad \sigma_n^2 = 2H_n - 4H_n^{(2)} + 2,$$

*and probability distribution*

$$\Pr\{D_n = k\} = \frac{2^k}{n\,!} \left[{n-1 \atop k}\right].$$

*The standard quadtree* ($d = 2$). In the case of dimension $d = 2$, the analytic model of quadtrees can be solved explicitly in terms of hypergeometric functions that are otherwise known to occur in the average case analysis of partial match.

**Theorem 3** *The cost $D_n$ of a random search in a standard quadtree of size $n-1$ has a generating function $\gamma_n(u)$ given by*

$$\gamma_n(u^2) \equiv \mathbf{E}\{u^{2D_n}\} = \frac{1}{4u^2 - 1} \sum_{j=0}^{n} \binom{2u}{j} \binom{2u-1}{j} \binom{2u-2+n-j}{n-j}.$$

Thus the probability distribution of $D_n$ is expressible as a complicated convolution of Stirling numbers.

**Proof.** The generating function $\Phi$ of the level polynomials satisfies

$$\Phi(u, z) = 1 + 2^2 u \int_0^z \frac{dx}{x(1-x)} \int_0^x \Phi(u, t) \frac{dt}{1-t}$$

an equation whose solution admits an hypergeometric form:

$$\Phi(u^2; z) = \frac{1}{(1-z)^{2u}} F[-2u, 1-2u; 1; z]. \tag{7}$$

where

$$F \equiv F[a, b; c; z] = 1 + \frac{a \cdot b}{c} \frac{z}{1!} + \frac{a(a+1) \cdot b(b+1)}{c(c+1)} \frac{z^2}{2!} + \cdots. \tag{8}$$

□

## 4 The singularity perturbation method

The architecture of the proof of the main theorem asserting asymptotic normality of the distribution is transparent although implementation of it requires quite some care. In essence, we need to solve a double inversion problem in order to recover the coefficients $[z^n u^k]\Phi(u,z)$. A first stage consists in extracting $\phi_n(u) = [z^n]\Phi(u,z)$. This involves examining the influence of a parameter ($u$) on the singularity (at $z = 1$) of a differential equation. For that reason we call our method a *singularity perturbation method*. It is intermediate in difficulty between regular perturbations and singular perturbations, the latter implying reduction of order[6]. A second stage (from $\phi_n(u)$ to its coefficients) relies on continuity theorems of analytic probability. We offer here a brief outline.
The starting point is the integral equation furnished by Lemma 1,

$$\Phi(u,z) = 1 + 2^d u \mathbf{J}^{d-1}\mathbf{I}\,\Phi(u,z). \tag{9}$$

That equation is itself equivalent to a linear differential equation with coefficients that are polynomial in the main variable $z$ and the parameter $u$. The order of the equation is equal to the dimension of the data space, $d$. The standard theory is more conveniently developed from *differential systems* rather than equations, and the associated system is also of dimension $d$.
The most common case for linear differential equations and systems is the one called *regular singularity* or *singularity of the first kind* [1, 12]. In such a case, a basis of solutions can be found that are of the approximate form $c/(1-z)^\alpha$. The possible exponents $\alpha$ are determined by substituting into the equation. They thus appear as roots of a polynomial called the *indicial polynomial*.
In a parameterized case like (9), we thus expect solutions to involve linear combinations of terms of the form

$$\frac{c(u)}{(1-z)^{\alpha(u)}}, \tag{10}$$

as $z \to 1$. In the case of (9), it is found that the exponents are the algebraic functions that are roots of the indicial equation

$$(\alpha(u))^d - 2^d u = 0.$$

Forms belonging to the general type (10) were already encountered when $d = 1$, see Eq. (6), and when $d = 2$, see Eq. (7).
As $z \to 1$, the dominant term in the expansion of $\Phi(u,z)$ is the one corresponding to the root $2u^{1/d}$ which has maximal real part. In particular when the parameter $u$ is close to 1, this is the principal determination of $2\sqrt[d]{u}$. From the shape (10) of singular elements, we thus expect the singular form of $\Phi$ to be

$$\Phi(u,z) \approx \frac{c(u)}{(1-z)^{2u^{1/d}}} \qquad (z \to 1), \tag{11}$$

at least for $u$ near 1.
According to the usual principles of singularity analysis, the *dominant singular behaviour* of $\Phi$ provides the dominant asymptotic term in its coefficients $\phi_n(u) = [z^n]\Phi(u,z)$. Translating (11) to coefficients, we expect to get, as an approximation of $\phi_n(u)$,

$$\phi_n(u) \approx c(u)\frac{n^{2u^{1/d}-1}}{\Gamma(2u^{1/d})}. \tag{12}$$

[6] We refer to standard texts like [1, 12] for basics on linear differential equations in the complex domain.

Given the approximation (12), values of the polynomial $\phi_n(u)$ are asymptotically known at least for $u$ in a neighbourhood of 1. An inversion problem (the second one after the phase of singularity analysis ensuring the transition from (11) to (12)) is then to be solved. The approximation (12) permits to estimate $\phi_n(e^{i\theta})$, suitably normalized, when $\theta$ lies near 0. The Fourier transform of the distribution defined by the coefficients of $\phi_n(u)$ tends to $e^{-\theta^2/2}$, the characteristic function of the Gaussian distribution:

$$\lim_{n\to+\infty} e^{-i\theta a_n/b_n}\ \frac{f_n(e^{i\theta/b_n})}{f_n(1)} = e^{-\theta^2/2},$$

for some suitably chosen centering constants $a_n, b_n$ (that may be taken equal to the mean $\mu_n$ and variance $\sigma_n$ of the distribution).
Since $\phi_n(u)$ has positive coefficients, the continuity theorem for characteristic functions (or equivalently Fourier transforms of measures) of analytic probability theory applies. This leads to the end result, namely the convergence in distribution to a normal distribution for the coefficients of $\phi_n(u)$ which in turn carries to the distribution of $D_n$ as expressed by the main theorem.
The method is expected to be instrumental for a wide class of recurrences whose generating functions satisfy parameterized linear differential equations.

## References


[1] E. A. Coddington and M. Levinson. *Theory of Ordinary Differential Equations.* McGraw-Hill, 1955.

[2] L. Devroye and L. Laforest. An analysis of random $d$–dimensional quad trees. *SIAM Journal on Computing*, 19:821–832, 1990.

[3] Ph. Flajolet, G. Gonnet, C. Puech, and J. M. Robson. Analytic variations on quadtrees. *Algorithmica*, 1992. 24 pages, to appear.

[4] Ph. Flajolet and Th. Lafforgue. Search costs in quadtrees and singularity perturbation asymptotics. In preparation, 1992.

[5] G. H. Gonnet and R. Baeza-Yates. *Handbook of Algorithms and Data Structures: in Pascal and C.* Addison–Wesley, second edition, 1991.

[6] M. Hoshi and Ph. Flajolet. Page usage in a quadtree index. *BIT*, 32:384–402, 1992.

[7] L. Laforest. Étude des arbres hyperquaternaires. Technical Report 3, LACIM, UQAM, Montreal, November 1990. (Author's PhD Thesis at McGill University).

[8] H. M. Mahmoud. *Evolution of Random Search Trees.* John Wiley, 1992.

[9] H. Samet. *Applications of Spatial Data Structures.* Addison–Wesley, 1990.

[10] H. Samet. *The Design and Analysis of Spatial Data Structures.* Addison–Wesley, 1990.

[11] R. Sedgewick. *Algorithms.* Addison-Wesley, Reading, Mass., second edition, 1988.

[12] W. Wasow. *Asymptotic Expansions for Ordinary Differential Equations.* Dover, 1987. A reprint of the John Wiley edition, 1965.

# 23

# Arbres digitaux et équations aux différences

Philippe Flajolet
INRIA, Rocquencourt

[résumé par Jean-Marc Steyaert]

Les arbres digitaux constituent une structure de données très riche qui apparaît dans plusieurs contextes d'application : arbres de recherche, stratégies de hachage dynamique, recherche multi-dimensionnelle, protocoles de réseaux, algorithmes probabilistes, etc. L'analyse de ces structures et algorithmes met en évidence trois modèles principaux que sous-tend la même structure d'arbre binaire :

I. les *tries* pour lesquels les sommets internes sont vides et les feuilles seules contiennent une clef ;

II. les *arbres de recherche digitaux* pour lesquels tous les sommets, internes et externes, contiennent une clef ;

III. les *arbres paginés* pour lesquels les sommets contiennent $b$ clefs pour les internes et jusqu'à $b$ clefs pour les feuilles.

La distribution du nombre de clefs dans les sous-arbres est dans les trois cas de type Bernoulli équilibré : la probabilité que le sous-arbre gauche contienne $p$ clefs et le sous-arbre droit $q$ clefs valant $\binom{p+q}{p}2^{-(p+q)}$. Les deux paramètres d'intérêt sont le nombre de sommets internes $S_n$(qui décrit la compacité) de l'arbre et la longueur de cheminement $L_n$(qui décrit le degré d'équilibrage) de l'arbre, dont on calcule les valeurs moyennes en fonction du nombre $n$ de clefs contenues dans l'arbre. Pour les trois modèles on attend des comportements respectifs du type $s_n = E(S_n) = O(n)$ et $l_n = E(L_n) = O(n \log n)$.
Bien que les résultats dus respectivement à Knuth-de Bruijn [1], Flajolet-Sedgewick [2] et Flajolet-Richmond [3] soient semblables, les méthodes mises en œuvre vont en se compliquant comme le laisse suggérer la forme des équations (différentielles) aux différences que satisfont les s.g.e. de dénombrement des divers paramètres :

$$\begin{aligned} \text{I.} \quad & f(z) &=& \ a(z) + 2e^{z/2} f(z/2), \\ \text{II.} \quad & \tfrac{d}{dz} f(z) &=& \ a(z) + 2e^{z/2} f(z/2), \\ \text{III.} \quad & \tfrac{d^k}{dz^k} f(z) &=& \ a(z) + 2e^{z/2} f(z/2). \end{aligned}$$

La forme de ces équations est bien sûr intimement liée au processus de Bernoulli. Montrons à titre d'exemple la formule de type I pour la taille des *tries*. Si $f_n$ est la valeur moyenne de la taille, elle satisfait la récurrence : $f_n = 1 - \delta_{n,0} + \sum_{p+q=n} \binom{n}{p} 2^{-p} 2^{-q} (f_p + f_q)$, où $\delta_{n,0}$ est le symbole de Kronecker. Ces récurrences se résolvent classiquement en passant aux s.g.e. et on obtient pour la s.g.e. $f(z) = \sum_{n \geq 0} f_n z^n / n!$ l'équation $f(z) = e^z - 1 + 2e^{z/2} f(z/2)$. Lorsqu'il l'on place une clef dans les sommets internes la sommation se fait pour $p + q = n - 1$, et il est bien connu que pour "compenser" il faut dériver ce qui donne le type II, et semblablement pour le type III.

## 1 Les *tries*

La résolution algébrique des équations de type I se fait par itération et on obtient en toute généralité : $f(z) = \sum_{k\geq 0} 2^k a(z/2^k)\exp(z(1-2^{-k}))$. Si $a(z)$ a une forme simple, l'extraction des coefficients de la série est routinière.

**Théorème 1** [Knuth] : *Les valeurs moyennes $s_n$ de la taille et $l_n$ de la longueur de cheminement des* tries *ont les expressions exactes :*

$$\begin{aligned} s_n &= \sum_{k\geq 0} 2^k(1-(1-2^{-k})^n - \frac{n}{2^k}(1-2^{-k})^{n-1}), \\ l_n &= n\sum_{k\geq 0}(1-(1-2^{-k})^{n-1}). \end{aligned}$$

L'asymptotique de ces sommes peut se faire de manière élémentaire si l'on se contente de l'ordre de grandeur. On observe en effet que les termes valent essentiellement 1 pour $k$ petit et 0 pour $k$ grand, avec une transition relativement brutale pour $k \sim \log_2 n$. On est déduit alors simplement : $s_n = O(n)$ et $l_n = n\log_2 n + O(n)$.
De façon plus fine, en utilisant une technologie classique en théorie analytique des nombres, Knuth et de Bruijn ont mis en évidence le caractère fluctuant de ses quantités autour d'une valeur moyenne dont l'ordre de grandeur a été obtenu précédemment.

**Théorème 2** [Knuth et de Bruijn] : *Les valeurs moyennes $s_n$ de la taille et $l_n$ de la longueur de cheminement des* tries *ont les expressions asymptotiques :*

$$\begin{aligned} 1+2s_n &= \frac{2n}{\log 2} + nQ(\log_2 n) + O(\sqrt{n}), \\ l_n &= n\log_2 n + nP(\log_2 n) + O(\sqrt{n}), \end{aligned}$$

*où $P$ et $Q$ sont deux fonctions périodiques de période* 1.

La méthode requise pour évaluer les sommes telles que $s_n$ et $l_n$, que nous appellerons *harmoniques*, repose sur la transformation intégrale de Mellin :

$$f^\star(s) = \int_0^\infty f(x)x^{s-1}dx,$$

qui associe donc à une fonction $f$ ayant de bonnes propriétés de croissance à l'origine ($f(x) = O(x^\alpha)$) et à l'infini ($f(x) = O(x^\beta)$), la fonction $f^\star(s)$ qui est analytique dans la bande $-\alpha < \Re(s) < -\beta$. La formule inverse de Perron

$$f(x) = \frac{1}{2i\pi}\int_{c-i\infty}^{c+i\infty} f^\star(s)x^{-s}ds, \text{ pour } -\alpha < c < -\beta,$$

permet souvent après étude des singularités de $f^\star$ d'en déduire le comportement asymptotique de $f(x)$, quand $x$ tend vers l'infini.
Illustrons l'ensemble du processus dans le cas de la longueur de cheminement $L_n$ d'un arbre de taille $n$ qui vérifie comme variable aléatoire dépendant de $x$, nombre de feuilles du sous-arbre gauche, la récurrence :

$$\begin{aligned} L_n &= n + L_x + L_{n-x}, \\ L_0 &= L_1 = 0. \end{aligned}$$

Comme $\mathbf{P}(x = k|n \text{ clefs}) = \pi_{nk} = 2^{-n}\binom{n}{k}$, on en déduit que l'espérance de $L_n$ satisfait la récurrence :

$$l_n = n - \delta_{n1} + \sum_{k=0}^{n} 2^{-n} \binom{n}{k} (l_k + l_{n-k}).$$

La résolution d'une telle récurrence se fait de manière (maintenant) classique en utilisant les s.g.e., $f_n \mapsto f(z) = \sum f_n z^n/n!$, et les constructions admissibles :

$$\begin{aligned} 1 &\mapsto e^z, \\ n &\mapsto ze^z, \\ \delta_{n0} &\mapsto 1, \\ \delta_{n1} &\mapsto z, \\ \frac{f_n}{2^n} &\mapsto f(z/2), \\ \sum_{k=0}^{n} \binom{n}{k} f_k &\mapsto e^z f(z). \end{aligned}$$

Dans ces conditions, il est aisé de déduire que la s.g.e. $l(z)$ des $l_n$ vérifie l'équation fonctionnelle :

$$l(z) = z(e^z - 1) + 2e^{z/2} l(z/2),$$

qui se résout par itération en

$$l(z) = \sum_{k\geq 0} 2^k e^{z(1-\frac{1}{2^k})} \frac{z}{2^k} (e^{z(1-\frac{1}{2^k})} - 1).$$

Après simplification, l'extraction des coefficients de Taylor de $l(z)$ est un jeu d'enfant et fournit l'expression du théorème 1.
Passons maintenant à l'étude du comportement asymptotique de $l_n$. Un calcul d'approximation un peu fastidieux, fondé sur le fait que quand $k$ est supérieur à $\log_2 n$, $(1 - 2^{-k})^n \approx e^{-n/2^k}$, permet de remplacer l'expression exacte de $l_n/n$ par la somme harmonique approchée suivante :

$$\frac{l_n}{n} = \sum_{k\geq 0} (1 - e^{-(n-1)/2^k}) + o(1).$$

Le problème revient donc à estimer le comportement asymptotique de la fonction $\Lambda(x) = \sum_{k\geq 0}(1 - e^{-x/2^k})$, lorsque $x$ tend vers l'infini.
Remarquons d'abord que la transformée de Mellin de $e^{-x}$ n'est autre que la bien connue fonction Gamma d'Euler : $\Gamma(s) = \int_0^\infty e^{-x} x^{s-1} dx$. La transformée de Mellin $\lambda^\star(s)$ de $\lambda(x) = 1 - e^{-x}$, est également $\Gamma(s)$, mais prise dans la bande $-1 < \Re(s) < 0$ (c 'est un des moyens de prolonger analytiquement $\Gamma(s)$). Par changement de variable trivial et linéarité, on a alors :

$$\Lambda^\star(s) = -\sum_{k\geq 0} 2^{ks} \Gamma(s) = \frac{-\Gamma(s)}{1 - 2^s},$$

fonction qui doit être considérée dans la bande fondamentale $-1 < \Re(s) < 0$. Le développement asymptotique pour $\Lambda(x)$, quand $x$ tend vers l'infini, est obtenu en appliquant à $\Lambda^\star(s)$ la formule de

Perron sus-mentionnée, dans la bande fondamentale , puis pour calculer cette intégrale en déplaçant l'abscisse d'intégration vers la droite tout en collectant les contributions de chacun des pôles de $\Lambda^\star(s)$ rencontrés au moyen de la formule des résidus : on utilise ici le fait que l'intégrande tend vers 0 lorsque la partie imaginaire de $s$ tend vers l'infini. Le fait que $\Lambda^\star(s)$ possède un pôle double en 0 (celui de $\Gamma(s)$ et de $(1-2^s)^{-1}$) et des pôles simples en $s = \frac{\pm 2ik\pi}{\log 2}$, $k > 0$, permet de conclure que :

$$\Lambda(x) = \frac{\log x}{\log 2} + \frac{1}{2} + \frac{\gamma}{\log 2} - \sum_{k\neq 0} \frac{\Gamma(\frac{2ik\pi}{\log 2})}{\log 2} x^{-\frac{2ik\pi}{\log 2}},$$

la dernière somme mettant en évidence la périodicité de la fonction $P$ du théorème 2 (et sa décomposition en série de Fourier), qui s'obtient immédiatement.

## 2 Les arbres digitaux

Ces arbres ont été étudiés par Knuth, Konheim et Newman pour ce qui est de leur longueur de cheminement et par Flajolet et Sedgewick pour ce qui est du nombre de sommets internes-externes, c'est-à-dire en frange de l'arbre.

**Théorème 3 :** *Les valeurs moyennes de la longueur de cheminement interne et du nombre de sommets internes-externes sont respectivement :*

$$(n+1)\log n + \frac{\gamma - 1}{\log 2} + \frac{1}{2} - \alpha + W(\log_2 n) + O(\sqrt{n})$$

*et*

$$\lambda.n + Q(\log_2 n).n + O(\sqrt{n}),$$

*où* $\alpha = 1 + 1/3 + 1/7 + 1/15 + 1/31 + \ldots$ *et* $W(u)$ *et* $Q(u)$ *sont périodiques de période 1.*

De fait $Q(u)$ a la forme surprenante suivante, $Q(u) = (1-u/2)(1-u/4)(1-u/8)\ldots$
Comme indiqué dans l'introduction, les quantités ci-dessus se définissent récursivement sur les sous-arbres gauche et droit, et dans ce cas le fait qu'une clef soit rangée dans chaque sommet interne (il y en a donc $n$) induit des récurrences légèrement différentes, du type :

$$f_n = a_n + \sum_{k=0}^{n} \pi_{nk}(f_k + f_{n-1-k}),$$

avec $\pi_{nk} = 2^{-(n-1)}\binom{n-1}{k}$, puisque seules $n-1$ clefs restent à placer.
Un calcul algébrique classique conduit ainsi aux équations différentielles aux différences de type II. Le schéma de résolution consiste à passer par les séries génératrices de Poisson : $g(z) = e^{-z}f(z) = \sum f_n e^{-z} z^n/n! = \sum_n g_n z^n/n!$. L'équation générique de type II se transforme alors en :

$$g'(z) + g(z) = \alpha(z) + 2g(z/2).$$

Ainsi a-t-on pour les coefficients la récurrence :

$$g_{n+1} + g_n = \alpha_n + 2^{1-n} g_n,$$

qui dans le cas de la longueur de cheminement, $a(z) = ze^z$, $\alpha(z) = z$, conduit à la formule explicite $g_n = (-1)^n \prod_{j=1}^{n-2}(1-2^{-j})$, puis par inversion à la forme explicite pour l'espérance de la longueur de cheminement :

$$E(l_n) = \sum_{k=2}^{n} \binom{n}{k} (-1)^k Q_{k-2},$$

avec $Q_m = (1-1/2)(1-1/4)(1-1/8)\ldots(1-1/2^m)$.
On est donc en présence d'une différence dont il faut faire l'asymptotique. La première remarque est que pour la plus part des fonctions, $u^m$, $\sqrt{u}$, $\log u$, $1/(u^2+1)$, $1/(1-2^u)$, etc. ces différences (qui s'apparentent à des dérivées de très grand ordre) sont faibles.
La clef de leur asymptotique est fournie par les intégrales de Rice :

$$S_n = \sum_{k=0}^{n} \binom{n}{k} (-1)^k f(k) = \frac{(-1)^n}{2i\pi} \int_{\mathcal{C}} f(z) \frac{n!}{z(z-1)(z-2)\ldots(z-n)} dz,$$

où $\mathcal{C}$ est un contour qui entoure simplement les points $0, 1, 2, \ldots, n$.
Si de plus $f$ ne croît pas trop vite à l'infini dans le demi-plan droit, il est possible de remplacer l'intégrale de contour par une sommation le long d'une droite imaginaire

$$(-1)^n \Delta^n f = \frac{1}{2i\pi} \int_{c-i\infty}^{c+i\infty} f(z) \frac{n!}{z(z-1)(z-2)\ldots(z-n)} dz,$$

puis si $f$ est méromorphe d'appliquer la technique déjà vue dans le cas précédent de balayer les pôles, vers la gauche cette fois, en collectant les résidus :

$$(-1)^n S_n = \sum Res f(z) \frac{n!}{z(z-1)(z-2)\ldots(z-n)} dz,$$

chaque pôle $\sigma$ contribuant ainsi pour $(Res_\sigma f)\dfrac{n^\sigma}{\Gamma(\sigma)}$ lorsqu'il est simple, $(Res_\sigma f)\dfrac{n^\sigma \log n}{\Gamma(\sigma)}$ lorsqu'il est double, etc.
Il faut donc extrapoler la fonction $f(k)$ au plan complexe, ce qui se fait par des méthodes à la Weierstrass. Dans le cas de la longueur de cheminement, on définira ainsi $Q(u) = \prod_{k>0}(1-u/2^k)$, de telle sorte que $f(z) = Q(1)/Q(2^{-z})$, dont les pôles (à une translation près) sont en $k \pm 2il\pi$, $k \leq 0$, ce qui conduit aux formules du théorème 3.

## 3 Les arbres digitaux paginés

Dans ce cas traité par Flajolet et Richmond, on place dans chaque nœud jusqu'à $b$ clefs. Le cas $b = 0$ correspond donc aux *tries* et le cas $b = 1$ aux arbres digitaux. La méthode de résolution précédemment utilisée conduit à des équations de type III sur la s.g.e, qui conduisent par passage aux séries génératrices de Poisson à des récurrences malheureusement non linéaires dès que $b > 1$. Le résultat principal donne un comportement du type déjà observé pour le nombre de sommets de la structure.

**Théorème 4** [Flajolet, Richmond] : *Le nombre moyen de sommets dans les arbres b-paginés vaut*

$$s_n = n(q_0 + R(\log_2 n)) + O(\sqrt{n}),$$

*où* $q_0 = \dfrac{1}{\log 2}\displaystyle\int_0^\infty \frac{(1+t)^{b-1}dt}{((1+t)(1+t/2)(1+t/4)\ldots)^b}$, *et* $R(u)$ *est périodique et développable en série de Fourier.*

Pour $b=2$, $q_0$ est une $q$-fonction de Bessel ; quand $b$ tend vers l'infini, $q_0 \approx 1/(b\log 2)$ ; et il est bien sûr toujours possible d'estimer numériquement $q_0$ : on observe que le taux moyen de remplissage est de l'ordre de 70 %.

La méthode est aussi fondée sur le passage aux séries de Poisson, mais cette fois sur les séries génératrices *ordinaires* ! On a alors, en notant $F(z)$ et $G(z)$ les s.g.o. de $f_n$ et $g_n$, les correspondances suivantes :

$$f_n = \sum_{k=0}^{n} \binom{n}{k} g_k \iff F(z) = \frac{1}{1-z} G(\frac{z}{1-z}),$$

et (bien sûr)

$$f_n = g_{n-b} \iff F(z) = z^b G(z).$$

La récurrence des $f_n$ se transpose heureusement sur les $g_n$ comme le montre l'énoncé suivant.

**Proposition :** *La s.g.o.* $F(z)$ *du nombre moyen de sommets dans les arbres b-paginés vaut*

$$F(z) = \frac{1}{1-z} G(\frac{z}{1-z}),$$

*où* $G(t)$ *est donné par*

$$G(t).(1+t)^b = P(t) + 2t^b G(t/2)$$

*avec* $P(t) = t(1+t)^{b-1}$.

On dispose alors d'une forme explicite pour $G(t)$ :

$$G(t) = \frac{P(t)}{(1+t)^b} + \frac{2t^b}{1+t^b}\frac{P(t/2)}{(1+t/2)^b} + \frac{2^2 t^b (t/2)^b}{(1+t)(1+t/2)^b}\frac{P(t/4)}{(1+t/4)^b} + \ldots,$$

ou encore, pour retrouver une forme déjà utilisée,

$$G(t) = \sum_{j\geq 0} 2^j P(2^j t)(\frac{Q(t/2)}{Q(2^j t)})^b, \qquad \text{avec} Q(u) = (1+u)(1+u/2)(1+u/4)...$$

Il reste alors à faire l'analyse du comportement de $G(t)$ pour $t = z/(1-z)$ tendant vers l'infini. Cette analyse se fait de nouveau par transformation de Mellin et conduit aux expressions du théorème 4.

## References


[1] N. G. De Bruijn, D. E. Knuth, and S. O. Rice. The average height of planted plane trees. In R. C. Read, editor, *Graph Theory and Computing*, pages 15–22. Academic Press, 1972.

[2] P. Flajolet and R. Sedgewick. Digital search trees revisited. *SIAM Journal on Computing*, 15(3):748–767, August 1986.

[3] Ph. Flajolet and B. Richmond. Generalized digital trees and their difference–differential equations. *Random Structures and Algorithms*, 3(3):305–320, 1992.

[4] D. E. Knuth. *The Art of Computer Programming*, volume 3 : Sorting and Searching. Addison-Wesley, 1973.

[5] A. G. Konheim and D. J. Newman. A note on growing binary trees. *Discrete Mathematics*, 4:57–63, 1973.

# 24

# Multidimensional Digital Searching

Helmut Prodinger
Technical University of Vienna

[résumé par Danièle Gardy]

Cet exposé présente les performances de la recherche partiellement spécifiée, sur des clés multidimensionnelles stockées dans des *tries*, ou dans d'autres structures digitales.

## 1 Présentation

Les *"tries"* (arbres lexicographiques, arbres préfixes) sont habituellement construits à partir de clés qui sont des mots infinis sur l'alphabet $\{0,1\}$; les nœuds internes servent à se diriger dans l'arbre et les clés sont stockées aux feuilles; on ne garde des clés que ce qui suffit à les distinguer des autres clés lors de l'insertion. On va s'intéresser dans cet exposé à leurs performances, lorsqu'on les étend à des clés multidimensionnelles.
*Structures m-dimensionnelles :* Ici, les clés sont des ensembles de $m$-uplets : $K = (K_1, K_2, \cdots, K_m)$, où les $K_i$ sont des suites de bits. Pour construire la structure arborescente, on remplace d'abord chaque clé composée $K$ par une chaîne de bits obtenue comme suit : on prend le bit initial de chaque composante $K_i$ (dans l'ordre), puis le deuxième bit, etc. Ensuite, on construit le *trie* sur la chaîne de bits, comme dans le cas unidimensionnel.
Le cas le plus simple est obtenu lorsque $m = 2$; c'est celui qui sera traité en détail dans la suite. Par exemple, pour $m = 2$, la clé $K = (01\ldots, 00\ldots)$ donne la chaîne $0010\ldots$, et la clé $(10\ldots, 01\ldots)$ donne la chaîne $1001\ldots$ Une *recherche partiellement spécifiée* consiste à chercher toutes les clés dont certaines coordonnées sont égales à une valeur spécifiée, les autres coordonnées pouvant prendre n'importe quelle valeur. Le profil (*search pattern*) est un mot de $\{*, S\}^m$ : $S$ correspond à une coordonnée spécifiée, et $*$ à une coordonnée non spécifiée. Une mesure de complexité classique pour ce problème est le nombre de nœuds internes visités. Attention, certains nœuds internes peuvent être visités même s'ils conduisent à une feuille contenant une clé qui ne répond pas à la requête.

## 2 Etude détaillée du cas m=2

Soit le profil $\omega = (*, S)$ : on cherche donc une clé dont la première composante est quelconque, et la seconde égale à une valeur donnée. Soit $\omega' = (S, *)$ le profil se déduisant de $\omega$ par permutation circulaire.

### 2.1 Coût moyen d'une recherche suivant le profil $\omega$

Notons $H_n^\omega(z)$ la fonction génératrice de probabilité du coût de la requête de profil $\omega$ sur des données de taille $n$ (i.e. le *trie* est construit à partir de $n$ clés bidimensionnelles) :

$$H_n^\omega(z) = \sum_{i \geq 0} \text{Proba}(\omega \text{ visite } i \text{ nœuds internes})\ z^i.$$

On définit de même $H_n^{\omega'}(z)$. Remarquons au passage que $H_n^{\omega}(1) = H_n^{\omega'}(1) = 1$.
Lors d'une recherche partiellement spécifiée dans un *trie*, et si la première coordonnée n'est pas spécifiée, la recherche se poursuit dans les deux sous-arbres; par contre, lorsque la première coordonnée est spécifiée, on ne poursuit la recherche que dans un seul sous-arbre (le gauche ou le droit, suivant la valeur qui est spécifiée). Dans les deux cas, la recherche au niveau inférieur se fait suivant le motif obtenu en décalant d'un bit le motif initial.
Si l'on suppose que les bits 0 et 1 ont même probabilité (hypothèse faite pour toute la suite), la probabilité que $k$ clés parmi les $n$ commencent par un bit donné est $\binom{n}{k}2^{-n}$. On obtient donc les équations de récurrence suivantes, liant le coût de la recherche suivant le motif $\omega = (*, S)$ au coût de la recherche suivant le motif permuté $\omega' = (S, *)$ :

$$\begin{aligned} H_n^{\omega}(z) &= z\sum_{k=0}^{n} 2^{-n}\binom{n}{k} H_k^{\omega'}(z)\, H_{n-k}^{\omega'}(z); \\ H_n^{\omega'}(z) &= z\sum_{k=0}^{n} 2^{-n}\binom{n}{k} H_k^{\omega}(z). \end{aligned}$$

De manière générale, si on travaille avec des clés de dimension $m$, on aura $m$ équations, chacune définissant la fonction génératrice pour un des profils obtenus par rotation du profil initial, à partir des $m-1$ autres fonctions génératrices associées aux profils permutés.
Revenons à l'exemple précédent : les valeurs moyennes $l_n^{\omega} = (H_n^{\omega})'(1)$ et $l_n^{\omega'} = (H_n^{\omega'})'(1)$, pour $n \geq 2$, satisfont les équations de récurrence suivantes, avec les conditions initiales $l_0^{\omega} = l_1^{\omega} = 1$ :

$$l_n^{\omega} = 1 + 2\sum_{k=0}^{n} 2^{-n}\binom{n}{k} l_k^{\omega'}, \qquad l_n^{\omega'} = 1 + \sum_{k=0}^{n} 2^{-n}\binom{n}{k} l_k^{\omega}. \tag{1}$$

Définissons les fonctions génératrices exponentielles de $l_n^{\omega}$ et de $l_n^{\omega'}$ : $L^{\omega}(z) = \sum_{n\geq 0} l_n^{\omega} z^n/n!$ et $L^{\omega'}(z) = \sum_{n\geq 0} l_n^{\omega'} z^n/n!$, puis les fonctions génératrices exponentielles de Poisson $\tilde{L}^{\omega}(z) = e^{-z}L^{\omega}(z)$ $= \sum_{n\geq 0} \tilde{l}_k^{\omega} z^n/n!$ et $\tilde{L}^{\omega'}(z) = e^{-z}L^{\omega'}(z) = \sum_{n\geq 0} \tilde{l}_k^{\omega'} z^n/n!$. Les coefficients de $L^{\omega}$ et $\tilde{L}^{\omega}$ sont liés par : $l_n^{\omega} = \sum_{k=0}^{n} \binom{n}{k}\tilde{l}_k^{\omega}$; de même ceux de $L^{\omega'}$ et de $\tilde{L}^{\omega'}$.
Les équations (1) sur les $l_n^{\omega}$ se traduisent d'abord en un système d'équations fonctionnelles sur les fonctions génératrices exponentielles $L^{\omega}$ et $L^{\omega'}$, puis en un système analogue sur les fonctions génératrices de Poisson :

$$\begin{aligned} \tilde{L}^{\omega}(z) &= 1 - (1+z)e^{-z} + 2\tilde{L}^{\omega'}(z/2); \\ \tilde{L}^{\omega'}(z) &= 1 - (1+z)e^{-z} + \tilde{L}^{\omega}(z/2). \end{aligned}$$

On passe alors à un système sur $\tilde{l}_n^{\omega}$ et $\tilde{l}_n^{\omega'}$, qu'on peut résoudre explicitement. Ainsi, on obtient :

$$\tilde{l}_n^{\omega} = (-1)^n \frac{(n-1)(1+2^{1-n})}{1-2^{1-2n}},$$

ce qui donne la moyenne $l_n^{\omega}$ sous forme de somme :

$$l_n^{\omega} = 1 + \sum_{k=1}^{n} \binom{n}{k}(-1)^k f(k), \qquad \text{avec} \qquad f(k) = \frac{(k-1)(1+2^{1-k})}{1-2^{1-2k}}. \tag{2}$$

La fonction $f$, définie sur les entiers positifs, peut être prolongée en une fonction analytique en posant $f(z) = (z-1)(1+2^{1-z})/(1-2^{1-2z})$. On peut alors utiliser la méthode de Rice, qui sert justement à calculer les sommes de la forme ci-dessus (voir par exemple la présentation qui en est faite dans [2, p. 754]). On a :

$$\sum_{k=1}^{n} \binom{n}{k} (-1)^k f(k) = -\frac{1}{2i\pi} \int_C [N;z] f(z) dz,$$

avec

$$[N;z] = \frac{\Gamma(-z)\Gamma(n+1)}{\Gamma(n+1-z)} = \frac{(-1)^{n+1} n!}{z(z-1)\cdots(z-n)},$$

et où $C$ est une courbe dans le domaine d'analycité de $f$, entourant les points $1, 2, \cdots n$. La somme (2) vaut asymptotiquement $\sum [N;c] f(c)$, où $c$ décrit l'ensemble des résidus à gauche de la droite $\Re z = 1$ (il suffit d'appliquer la formule des résidus à l'intégrande, qui a pour seuls pôles les entiers positifs). D'après la définition de $f$, les seuls résidus sont obtenus pour $\Re z = 1/2$, plus précisément pour $z = 1/2(1+\xi_k)$ avec $\xi_k = 2ik\pi/\log 2$. Donc, en posant $L = \log 2$ :

$$l_n^\omega \approx \sqrt{n} \left( \sqrt{\pi} \frac{1+\sqrt{2}}{2L} + \tau^\omega(\log_2 \sqrt{n}) \right) \approx \sqrt{n} \left( 3,086705281... + \tau^\omega(\log_2 \sqrt{n}) \right), \qquad (3)$$

où $\tau^\omega$ est une fonction périodique, de moyenne 0 et de période 1, d'amplitude faible, et qui peut s'écrire comme somme d'une série de Fourier :

$$\tau^\omega(x) = \sum_{k \neq 0} \tau_k e^{2ik\pi x} \qquad \text{avec} \qquad \tau_k = \frac{1}{2L}(1+(-1)^k\sqrt{2}) \, \frac{-1+\xi_k}{2} \, \Gamma(\frac{-1-\xi_k}{2}). \qquad (4)$$

On peut transformer l'expression des coefficients $\tau_k$, en utilisant le fait que $\Gamma(z+1) = z\Gamma(z)$; on obtient :

$$\tau_k = \frac{1}{2L}(1+(-1)^k\sqrt{2}) \, \frac{1-\xi_k}{1+\xi_k} \, \Gamma(\frac{1-\xi_k}{2}).$$

## 2.2 Variance

Le calcul de la variance du coût d'une recherche suivant le profil $\omega$ fait intervenir $W_n^\omega = (H_n^\omega)''(1)$. Soient $W^\omega$ et $\tilde{W}^\omega$ la fonction génératrice exponentielle et la fonction génératrice de Poisson associées : $W^\omega(z) = \sum_{n \geq 0} W_n^\omega z^n/n!$ et $\tilde{W}^\omega(z) = e^{-z} W^\omega(z)$. On obtient les équations fonctionnelles suivantes :

$$\tilde{W}^\omega(z) = 2\tilde{W}^{\omega'}(z/2) + 4\tilde{L}^{\omega'}(z/2) + 2(\tilde{L}^{\omega'}(z/2))^2, \qquad \tilde{W}^{\omega'}(z) = \tilde{W}^\omega(z/2) + 2\tilde{L}^\omega(z/2).$$

Il est possible d'en déduire une formule explicite, assez compliquée, pour les $W_n^\omega$ :

$$W_n^\omega = \sum_{k=2}^{n} \binom{n}{k} (-1)^k f(k),$$

avec

$$\begin{aligned} f(k) \;=\; & \frac{(z-1)2^{1-z}}{(1-2^{1-2z})(1-2^{2-2z})} \left( 5 + 2^{3-z} + 2^{3-2z} + z2^{z-2} - 2^z + \frac{z}{4} + (z-\frac{9}{2}) \left(\frac{3}{2}\right)^{z-2} \right) \\ & + \frac{2^{3-2z}}{(1-2^{1-2z})(1-2^{2-2z})} \sum_{l \geq 2} \binom{z}{l} \frac{(l-1)(z-l+1)(1+2^{-l})(1+2^{z-l})2^{-l}}{1-2^{1-2l}}. \end{aligned}$$

On applique là aussi la méthode de Rice; les résidus sont en $1+\xi_k$, et on obtient une expression asymptotique pour $W_n^\omega$, de la forme suivante, pour des coefficients $\alpha$, $\beta$ et $\gamma$ calculables et donnés explicitement dans [5, formule (28)] :

$$W_n^\omega \approx \alpha n + \beta\sqrt{n} + \gamma. \tag{5}$$

La variance $\sigma^2$ vaut $W_n^\omega + l_n^\omega - (l_n^\omega)^2$. Compte tenu des approximations (3) et (5), elle est donc de la forme $\sigma^2 = Bn + A\sqrt{n} + o(\sqrt{n})$. On montre que $B = 0$ (voir la section suivante), et donc que $\sigma^2 \approx A\sqrt{n}$. La variance a alors pour terme principal $A\sqrt{n}$, où, comme pour l'expression asymptotique de la moyenne, $A$ est la somme d'un terme constant $A_1$ et d'un terme périodique, de moyenne 0 et d'amplitude faible. De plus, on a une forme explicite pour $A_1$, ce qui permet de le calculer numériquement : $A_1 = 2{,}0918454...$

### 2.2.1 Preuve que B=0

Le coefficient $B$ est égal à la somme d'un terme périodique et d'un terme constant $B_1$ :

$$B_1 = (1/L)\left(8 - \frac{7}{12} - 2\sum_{l\geq 2}(-1)^l \frac{(1+2^{-l})(1+2^{1-l})2^{-l}}{1-2^{1-2l}}\right) - \pi\frac{(1+\sqrt{2})^2}{4L^2} - [(\tau^\omega)^2]_0,$$

où $[(\tau^\omega)^2]_0$ est la moyenne du carré de $\tau^\omega$; la fonction périodique $\tau^\omega$ est celle définie dans (4). Quelques manipulations algébriques permettent d'exprimer ce dernier terme :

$$[(\tau^\omega)^2]_0 = \sum_{k\neq 0}\tau_k\tau_{-k} = 2\sum_{k\geq 1}(1+(-1)^k\sqrt{2})^2\,\Gamma(\frac{1-\xi_k}{2})\,\Gamma(\frac{1+\xi_k}{2}).$$

Le produit $\Gamma((1-\xi_k)/2)\Gamma((1+\xi_k)/2)$ peut être simplifié grâce à la formule des compléments $\Gamma(s)\Gamma(1-s) = \pi/\sin(\pi s)$; on montre ainsi qu'il est égal à $\pi/\cos(\pi\xi_k/2)$, ou encore à $2\pi/(e^{k\pi^2/L} + e^{-k\pi^2/L})$. On obtient finalement une expression de $[(\tau^\omega)^2]_0$ faisant intervenir les fonctions $F(x) = \sum_{k\geq 1} e^{-kx}/(1+e^{-2kx})$ et $G(x) = \sum_{k\geq 1}(-1)^{k-1}e^{-kx}/(1+e^{-2kx})$ :

$$[(\tau^\omega)^2]_0 = \frac{3\pi}{L^2}\,F\left(\frac{\pi^2}{L}\right) - \frac{2\sqrt{2}\pi}{L^2}G\left(\frac{\pi^2}{L}\right).$$

La fonction $F$ vérifie une équation fonctionnelle (qui sera prouvée dans la section 2.2.2) :

$$F(x) = \frac{\pi}{4x} - \frac{1}{4} + \frac{\pi}{x}F(\frac{\pi^2}{x}) \qquad (0 < x < \pi^2).$$

On obtient donc, en remarquant de plus que $G(x) = F(x) - 2F(2x)$ :

$$[(\tau^\omega)^2]_0 = \frac{3-2\sqrt{2}}{L}F(L) + \frac{2\sqrt{2}}{L}F\left(\frac{L}{2}\right) + \frac{3}{4L} - \frac{(3+2\sqrt{2})\pi}{4L^2}.$$

On en tire :

$$B_1 = \frac{20}{3L} + \frac{2\sqrt{2}-3}{L}F(L) - \frac{2\sqrt{2}}{L}F(\frac{L}{2}) - \frac{2}{L}\sum_{n\geq 2}(-1)^n\frac{(1+2^{-n})(1+2^{1-n})2^{-n}}{1-2^{1-2n}}. \tag{6}$$

Sur cette expression, on peut alors montrer que $B_1$ est nul (voir section 2.2.3). Donc, $B$ se réduit à un terme périodique, de moyenne nulle. Or ce terme périodique est lui aussi identiquement nul. Supposons en effet qu'il ne le soit pas : multiplié par $n$, il donnerait alors le terme principal de la variance, qui serait alors négative pour un nombre infini de valeurs de $n$.

### 2.2.2 Comment prouver l'équation fonctionnelle sur $F$?

On développe $F$ :

$$F(x) = \sum_{k\geq 1} \sum_{j\geq 0} (-1)^j e^{-k(2j+1)x}.$$

On a donc une somme harmonique, à laquelle on peut appliquer la transformation de Mellin (voir par exemple [8, p. 453-455] pour la définition et les propriétés de base de la transformée de Mellin en analyse d'algorithmes; on peut aussi regarder, entre autres, [6, p. 18 et suivantes]) :

$$\begin{aligned} F^*(s) &= \int_0^{+\infty} F(x)x^{s-1}dx = \sum_{k,j}(-1)^j \int_0^{+\infty} e^{-k(2j+1)x}x^{s-1}dx \\ &= \sum_{k,j}(-1)^j k^{-s}(2j+1)^{-s} \int_0^{+\infty} e^{-t}t^{s-1}dt = \Gamma(s) \sum_{k\geq 1} k^{-s} \sum_{j\geq 0}(-1)^j(2j+1)^{-s}. \end{aligned}$$

Or $\sum_{k\geq 1} k^{-s} = \zeta(s)$; en posant $\beta(s) = \sum_{j\geq 0}(-1)^j(2j+1)^{-s} = 1 - 1/3^s + 1/5^s - \cdots$ on obtient $F^*(s) = \Gamma(s)\,\zeta(s)\,\beta(s)$. Par ailleurs, $F(x) \leq e^{-x}/(1-e^{-x})$, et $F$ a donc comme bande fondamentale $< 1; +\infty >$. La formule d'inversion de Mellin donne alors, en intégrant sur la droite verticale d'équation $\Re(s) = 3/2$ contenue dans cette bande fondamentale :

$$F(x) = \frac{1}{2i\pi} \int_{3/2-i\infty}^{3/2+i\infty} \Gamma(s)\,\zeta(s)\,\beta(s)\,x^{-s}ds.$$

La fonction $\Gamma(s)$ a des pôles simples en $s = 0, -1, -2, \ldots$, et la fonction $\zeta(s)$ a un pôle simple unique en $s = 1$. En ce qui concerne la fonction $\beta$, on remarque d'abord qu'elle peut s'exprimer à l'aide de la fonction de Hurwitz $\zeta(a, s) = \sum_{n\geq 0}(n+a)^{-s}$ [9, p. 265] :

$$\beta(s) = \frac{\zeta(s, \frac{1}{4}) - \zeta(s, \frac{3}{4})}{4^s}.$$

Par ailleurs, la fonction $\zeta(s, a)$ est analytique pour tout $s$, sauf en $s = 1$ qui est un pôle simple [1, p. 255] [9, p. 266]. La fonction $\beta$, étant définie au point 1, est donc entière, et $\beta(1) = \pi/4$. Notons aussi que $\beta(0) = 1/2$ et $\zeta(0) = -1/2$. En déplaçant le contour d'intégration à gauche des pôles 0 et 1 (par exemple sur la droite verticale $\Re z = -1/2$), on ajoute les résidus de l'intégrande en ces pôles. Le résidu pour $s = 1$ donne $\Gamma(1)\beta(1)/x = \pi/(4x)$, et celui pour $s = 0$ donne $\beta(0)\zeta(0) = -1/4$. On obtient donc :

$$F(x) = \frac{\pi}{4x} - \frac{1}{4} + \frac{1}{2i\pi} \int_{-1/2-i\infty}^{-1/2+i\infty} \Gamma(s)\,\zeta(s)\,\beta(s)\,x^{-s}ds.$$

On utilise ensuite la formule de duplication de la fonction $\Gamma$ [7, p. 160] [9, p. 240] :

$$\Gamma(s) = \Gamma(\frac{s}{2})\Gamma(\frac{s+1}{2})\frac{2^{s-1}}{\sqrt{\pi}}, \tag{7}$$

puis [7, p. 160] :

$$\Gamma(\frac{s}{2})\zeta(s) = \pi^{s-1/2}\Gamma(\frac{1-s}{2})\zeta(1-s).$$

La fonction $\zeta(a,s)$ vérifie l'équation fonctionnelle suivante, pour tous $h$ et $k$ entiers tels que $1 \leq h \leq k$ : [1, p. 261] :

$$\zeta(1-s,\frac{h}{k}) = \frac{2\Gamma(s)}{(2\pi k)^s}\sum_{r=1}^{k}\cos(\frac{\pi s}{2}-\frac{2\pi rh}{k})\,\zeta(s,\frac{r}{k}).$$

Cette équation, la formule des compléments et la formule de duplication de la fonction $\Gamma$ permettent d'obtenir l'équation fonctionnelle reliant $\beta$ et $\Gamma$ :

$$\beta(1-s)\Gamma(1-\frac{s}{2}) = 2^{2s-1}\pi^{-s+1/2}\Gamma(\frac{s+1}{2})\beta(s).$$

On a alors

$$\Gamma(s)\,\zeta(s)\,\beta(s) = 2^{-s}\pi^{2s-3/2}\,\Gamma(\frac{1-s}{2})\,\Gamma(1-\frac{s}{2})\,\zeta(1-s)\,\beta(1-s),$$

et il suffit d'appliquer la formule (7) à $\Gamma(1-s)$ pour obtenir

$$\Gamma(s)\,\zeta(s)\,\beta(s) = \pi^{2s-1}\,\Gamma(1-s)\,\zeta(1-s)\,\beta(1-s).$$

L'expression intégrale de $F(x)$ devient alors

$$F(x) = \frac{\pi}{4x} - \frac{1}{4} + \frac{1}{2i\pi}\int_{(-1/2)}\pi^{2s-1}\Gamma(1-s)\zeta(1-s)\beta(1-s)x^{-s}ds.$$

Un changement de variable $1-s=u$ permet de conclure :

$$\int_{(-1/2)}\pi^{2s-1}\Gamma(1-s)\zeta(1-s)\beta(1-s)x^{-s}ds = \frac{\pi}{x}\int_{(3/2)}\Gamma(u)\zeta(u)\beta(u)(\frac{x}{\pi^2})^u du,$$

et donc $F(x) = \pi/(4x) - 1/4 + (\pi/x)F(\pi^2/x)$.
On a donc prouvé l'équation fonctionnelle sur $F$. Ce type d'analyse revient fréquemment dans les analyses de *tries* : pour montrer la nullité d'un coefficient, on est amené à établir des identités fonctionnelles du type $f(x) =< expr > + f(C\pi^2/x)$. A titre d'exemple, voici quelques identités, se prouvant de la même manière que ci-dessus :

$$\begin{aligned}
f(x) &= \textstyle\sum_{k\geq 1}\frac{k}{e^{kx}-1} &&= \frac{\pi^2}{6x^2} - \frac{1}{2x} + \frac{1}{24} - \frac{4\pi^2}{x^2}f(\frac{4\pi^2}{x});\\
g(x) &= \textstyle\sum_{k\geq 1}\frac{(-1)^{k-1}}{k(e^{kx}-1)} &&= \frac{\pi^2}{12x} - \frac{\log 2}{2} + \frac{x}{24} - g(\frac{2\pi^2}{x});\\
h(x) &= \textstyle\sum_{k\geq 1}\frac{1}{k(e^{kx}-1)} &&= \frac{\pi^2}{6x} - \frac{x}{24} + \frac{\log(2\pi x)}{2} + h(\frac{4\pi^2}{x}).
\end{aligned}$$

### 2.2.3 Nullité de $B_1$

L'équation (6), légèrement transformée, donne :

$$B_1\;L = \frac{20}{3} + 2\sqrt{2}\left(F(L) - F(\frac{L}{2})\right) - 3F(L) - 2\sum_{n\geq 2}(-1)^n\frac{(1+2^{-n})(1+2^{1-n})2^{-n}}{1-2^{1-2n}},$$

avec $F(x) = \sum_{k\geq 1} e^{-kx}/(1+e^{-2kx})$. On a donc, pour $L = \log 2$ :

$$F(L) = \sum_{k\geq 1}\frac{2^{-k}}{1+2^{-2k}}; \qquad \text{et} \qquad F(\frac{L}{2}) = \sum_{k\geq 1}\frac{2^{-k/2}}{1+2^{-k}}.$$

On décompose la somme qui intervient dans $F(L/2)$, suivant la parité de $k$, et on obtient :

$$F(L/2) = F(L) + \sqrt{2} \sum_{p \geq 1} \frac{2^{-p}}{1 + 2^{1-2p}},$$

ce qui montre que

$$B_1\ L = \frac{20}{3} + 4 \sum_{p \geq 1} \frac{2^{-p}}{1 + 2^{1-2p}} - 3F(L) - 2 \sum_{n \geq 2} (-1)^n \frac{(1 + 2^{-n})(1 + 2^{1-n}) 2^{-n}}{1 - 2^{1-2n}}.$$

Par ailleurs,

$$\begin{aligned} F(L) &= \textstyle\sum_{k \geq 1} 2^{-k} \sum_{j \geq 0} (-1)^j 2^{-2kj} &&= \displaystyle\sum_{j \geq 0} (-1)^j \sum_{k \geq 1} 2^{-k(2j+1)} \\ &= \textstyle\sum_{j \geq 0} (-1)^j \frac{2^{-1-2j}}{1 - 2^{-1-2j}} &&= \displaystyle\sum_{j \geq 1} (-1)^{j+1} \frac{2^{1-2j}}{1 - 2^{1-2j}}. \end{aligned}$$

Enfin,

$$\frac{2^{-n}(1 + 2^{-n})(1 + 2^{1-n})}{1 - 2^{1-2n}} = -2^{-n} + \frac{2^{1-n}}{1 - 2^{1-2n}} + 3 \frac{2^{-2n}}{1 - 2^{1-2n}}.$$

En reportant ceci et la dernière expression de $F$ dans $B_1$, on remarque que la plupart des sommes s'annulent, et on trouve : $B_1\ L = 2/3 + 2 \sum_{n \geq 1} (-1)^n 2^{-n} = 0$.

### 2.3 Profil permuté $\omega' = (S, *)$

Son coût moyen cède au même type d'analyse, et on obtient, avec encore une fois des fonctions $\tau_1$ et $\tau_2$ périodiques, de moyenne 0 et de période 1, et de faible amplitude :

$$\begin{aligned} l_n^{\omega'} &\approx \sqrt{n} \left( \sqrt{\pi} \frac{1 + \sqrt{2}}{2\sqrt{2} \log 2} + \tau_1(\log_2 \sqrt{n}) \right) \approx \sqrt{n}\ (2,182630235... + \tau_1(\log_2 \sqrt{n}))\ ; \\ \sigma^2 &\approx \sqrt{n}\ (1,15324222... + \tau_2(\log_2 \sqrt{n}))\ . \end{aligned}$$

## 3 Généralisations et extensions

### 3.1 Autres types de données

La méthode exposée dans la section 2 permet d'analyser les performances de la recherche multidimensionnelle sur d'autres structures : arbres Patricia, arbres de recherche digitaux. Les versions multidimensionnelles de ces structures sont obtenues en construisant comme à l'accoutumée les arbres, sur des clés infinies obtenues à partir des clés composées. Par contre, on n'a que les valeurs moyennes des coûts de recherche, et non les variances.

Notons $l_{\omega,n}^{[A]}$ le nombre moyen de nœuds visités, pour une recherche suivant le profil $\omega$ dans un arbre de type $A$ et avec $n$ clés ($A$ vaut $D$ pour un arbre de recherche digital, $P$ pour un arbre Patricia, et $T$ pour un *trie*). On peut alors obtenir le coût moyen d'une recherche partiellement spécifiée : lorsque $s$ attributs parmi $m$ sont spécifiés, le coût est d'ordre $n^{1-s/m}$ :

$$l_{\omega,n}^{[A]} \approx C^{[A]} n^{1-s/m}.$$

La constante $C^{[A]}$ diffère selon le type de la structure :

$$\begin{aligned} C^{[P]} &= (1+\frac{s}{m})\,(1-2^{-s/m})\,\frac{1}{mL}\,\frac{\Gamma(s/m)}{1-s/m}\,\mathcal{S}; \\ C^{[D]} &= \frac{Q(2^{1-s/m})}{Q_\infty}\,\frac{1}{mL}\,\frac{\Gamma(s/m)}{1-s/m}\,\mathcal{S}; \\ C^{[T]} &= \frac{s}{m^2 L}\,\frac{\Gamma(s/m)}{1-s/m}\,\mathcal{S}. \end{aligned}$$

Dans ces formules, $\mathcal{S} = \sum_{j=0}^{m-1} \delta_1 \dots \delta_j 2^{-j(1-s/m)}$, et $\delta_i$ vaut 1 si le $i^{ème}$ attribut est spécifié, et 0 sinon. Enfin, la fonction $Q$ est définie par : $Q(x) = (1-x/2)(1-x/4)(1-x/8)\cdots$, et $Q_\infty = Q(1) = 0.28878809...$ Le résultat sur les *tries* avait été obtenu par Flajolet et Puech [4]. On peut comparer les constantes (prendre $x = s/m < 1$, et voir que $(1+x)(1-2^{-x}) < Q(2^{1-x})/Q_\infty < x$) :

$$C^{[P]} < C^{[D]} < C^{[T]}.$$

### 3.2 Autres types de problèmes

Des récurrences du type de (1) peuvent apparaître dans d'autres problèmes que ceux issus de l'étude d'une structure arborescente, ainsi le dénombrement probabiliste analysé dans [3].

## Références


[1] T. M. Apostol. *Introduction to Analytic Number Theory.* Springer-Verlag, 1976.

[2] P. Flajolet and R. Sedgewick. Digital search trees revisited. *SIAM Journal on Computing*, 15(3):748–767, August 1986.

[3] Ph. Flajolet and G. N. Martin. Probabilistic counting algorithms for data base applications. *Journal of Computer and System Sciences*, 31(2):182–209, October 1985.

[4] Ph. Flajolet and C. Puech. Partial match retrieval of multidimensional data. *Journal of the ACM*, 33(2):371–407, 1986.

[5] P. Kirschenhofer, H. Prodinger, and W. Szpankowski. Multidimensional digital searching and some new parameters in tries. Technical Report CSD-TR-91-052, Purdue University, July 1991.

[6] H. M. Mahmoud. *Evolution of Random Search Trees.* John Wiley, 1992.

[7] G. Tenenbaum. *Introduction à la théorie analytique et probabiliste des nombres.* Institut Élie Cartan, Nancy, 1992.

[8] J. Vitter and Ph. Flajolet. Analysis of Algorithms and Data Structures. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume A: Algorithms and Complexity, chapter 9, pages 431–524. North Holland, 1990.

[9] E. T. Whittaker and G. N. Watson. *A Course of Modern Analysis.* Cambridge University Press, fourth edition, 1927. Reprinted 1973.

# 25

# Compact Balanced Tries

Pierre Nicodème
Copernique and INRIA, Rocquencourt

[summary by Mireille Régnier]

Classical *B−trees* and *prefix B−trees* [1] offer both fast, direct addressing and easy sequential processing. They are balanced, segmented, and flexible. Flexibility means that a $B-$tree leaf splitting may be done at any position inside the leaf. This property is emphasised: one generates and suppresses empty leaves, while forcing the other leaves to a 100% storage utilisation. This property is important when memory utilisation is a crucial matter, as in memory databases. The segmentation allows parallel processing.
Other methods have been proposed in the last decade; they do not offer all the $B-$tree properties: the *bounding disorder method* [8], *Trie hashing* [7], *compact 0-complete tree* [10], the *compact trie* [5, 3] (section 1). This last structure is based on a bit-map representation of the tries. It is a simple, powerful, but not segmented structure.
It is shown (section 2) how to split the compact trie in a segmented and flexible structure of $B-$tree type: the *compact balanced trie*. Experimental results are given in section 3.

## 1 Compact Trie

The compact trie representation used by Kouacou-Kouadio, De Jonge, Tanenbaum and Van De Riet [5, 3] is composed of a bit-map and of a pointer-list. The 0 or 1 bit value of the trie corresponds to the digital values of the keys (Figure 1). The bit-map is the $0-1$ sequence obtained by right to left preorder traversal of the trie (Figure 1). The pointer list associates a pointer to each leaf of the trie. The basic property of this representation is that any bit of the bit-map followed by a 1-bit, as well as the last bit, represents a leaf node.
*Insertion* and *deletion* imply updating of the bit-map and of the pointer-list. The *retrieval* algorithm is based on a joint processing of the bit-map and of the pointer-list. Each bit of the retrieval key is checked from left to right; for each 1-bit found, the corresponding 0-subtrie is skipped over. This skipping is straightforward: checking the type (internal node or leaf) is easily done with the characterisation above and in any subtrie the number of leaves exceeds by 1 the number of internal nodes. The structure of Figure 2 is used for a compact representation. Each pointer is four bytes long, and the first byte of each pointer is the number of consecutive NIL-pointers at this point. The compact trie is not a segmented structure; therefore the linear search algorithm is $O\left(n^2\right)$ key comparisons with $n$ keys, and partial locking is impossible.

## 2 Compact-Balanced Tries

### 2.1 Trie Splitting

Figure 3 illustrates a trie splitting into two subtries. The subtrie $T1$ differs only from the trie $T$ by an incomplete bit-map and pointer-list. An edge-key and a corresponding edge-depth are added to

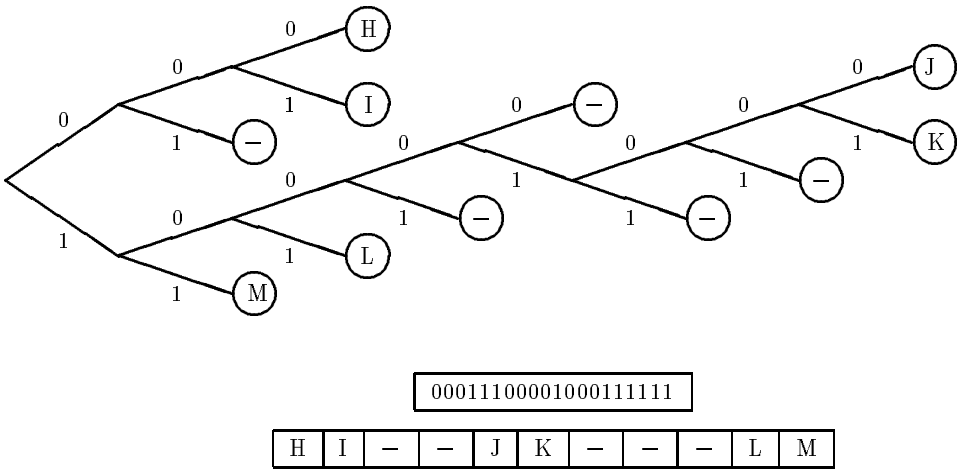



Figure 1: binary trie (bit-list and pointer-list)

| 20 | | 6 | | | |
|---|---|---|---|---|---|
| 00011100001000111111 | | | | | |
| 0 | H | 0 | I | 2 | J |
| 0 | K | 3 | L | 0 | M |

Figure 2: compact representation corresponding to the trie of Figure 1

subtrie $T2$. This allows (subsection 2.2) to restart the linear search algorithm of the compact-trie from the node corresponding to the edge key. Therefore, splitting a trie (or equivalently a subtrie) implies the splitting of the bit-map and of the pointer-list, and the creation of an edge-key, with a corresponding edge-depth.
There are *no constraints* for the choice of the splitting point. It may be chosen at the middle of the pointer-list which is the biggest part of the compact trie representation. The iterative splitting of the trie (and of the corresponding CB-nodes) generates a balanced tree structure.

## 2.2 Retrieval algorithm and edge-depth calculation

Shadow interior nodes and 0-leaves, i.e. leaves accessed through a 0-bit, must be handled along the edge during the retrieval algorithm. Key $F$ retrieval skips the subtrie containing the keys $A$, $C$, $D$, $E$, and four shadow nodes; this subtrie contains 4 interior nodes (3 shadow), and 5 leaves (1 shadow). The knowledge of the edge-key allows to rebuild the shadow nodes; the edge-subtrie skipping algorithm integrates this construction inside the ordinary subtrie skipping algorithm. The splitting process requires the computation of the depth of a new edge key, which corresponds to the computation of the depth of the corresponding node in the trie. This computation makes use of a stack algorithm (Figure 5 represents the stack evolution for trie T of Figure 3). The stack is built from the positions of the 0-bits of the nodes accessed in a preorder traversal of the trie, by

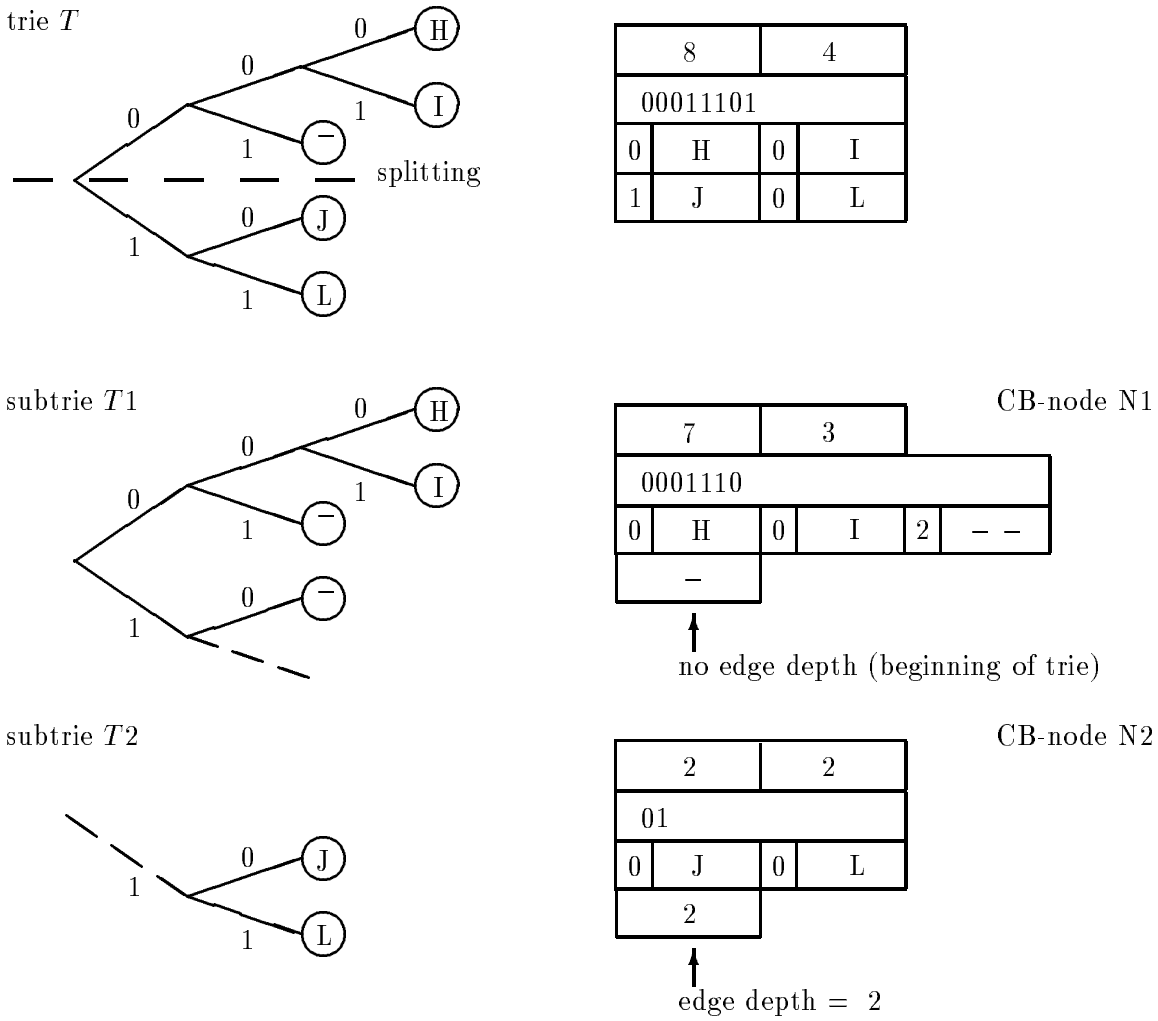



Figure 3: trie splitting and corresponding CB-nodes (compact-balanced nodes)

use of the bit-map.
The bit-map is scanned from left to right. For each 0-bit, the depth is increased by one and stacked. For a 1-bit, if the preceding bit is a zero, the depth remains unchanged, else the depth is equal to the depth of the top of the pile; thereafter unstacking is performed.
When proceeding with an edge key, the starting depth is the edge-depth and the pile has to be loaded with the 0-bit positions of the edge-key (limited to the edge-depth); then the node depth may be computed as previously. And the CB-Node splitting is achieved.

## 2.3 Merging and Balancing

The balancing process between two CB-nodes may be done by merging the 2 CB-nodes into a double size one, and then splitting this node into two CB-nodes; merging 2 CB-nodes $C1$ and $C2$ (keys of $C2$ being bigger than keys of $C1$) results in inserting the edge-key of $C2$ in $C1$, adjusting the bit-map and pointer-list of $C2$ and concatenating them to those of $C1$.

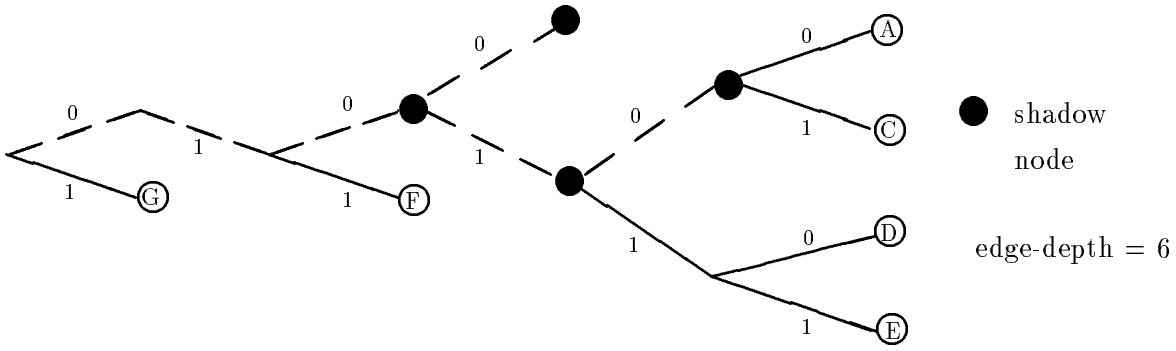



Figure 4: Edge-subtrie skipping (retrieve key $F$)

| index | value | depth | pile (0-bits) |
|---|---|---|---|
| 1 | 0 | 1 | 1 |
| 2 | 0 | 2 | 1, 2 |
| 3 | 0 | 3 | 1, 2, 3 |
| 4 | 1 | 3 | 1, 2 |
| 5 | 1 | 2 | 1 |
| 6 | 1 | 1 | - |
| 7 | 0 | 2 | 2 |
| 8 | 1 | 2 | - |

Figure 5: node depth computing

## 3 Experimental Results

Experimental results have been obtained on the Unix "words" dictionary and on an equivalent number of randomly distributed keys.

| Bits/Keys | number of keys | bit-map | NIL-pointers | compressed NIL-pointers |
|---|---|---|---|---|
| Sequential data | any | 2.0 | 8 | 1.0 |
| Random data | 25259 | 2.8 | 8 | 2.1 |
| Unix Dictionary | 25259 | 10.2 | 8 | 4.2 |

These results, (average number of bits per key in the bit-map and the NIL-pointers list), compare favorably to the results of the compact 0-complete tree (8 bits per key for random data with a 57% storage utilisation [11]); the storage utilisation of a compact balanced trie may be raised to 100%, thanks to the flexibility property (no constraints on the splitting point).

Sequential data do not generate NIL-leaves, while the high number of bits in the case of the Unix Dictionary is strongly dependent on the structure of alphanumeric data which generates many NIL-leaves.

## 4 Conclusions

These compact-balanced tries provide excellent compaction results. The relative moderate performance of the bit-map handling and the relative complexity of the algorithm are slight drawbacks. Suggestions (use of translation tables) are made in [3] to avoid bit-string handling.
It is proved that the linear representation of tries can be segmented and handled with a complete flexibility; this important novelty allows to step down from global linearity of the algorithms to local linearity and insures that worst cases of insertions are handled as well as a $B-$tree could do. The compact-balanced tries may be considered compact $B-$trees and could easily be implemented when lazy deletions policies are used [4]; their segmentation allows parallel partial locking and parallel processing [2], while their compactness fits with the constraints of memory databases (performances should be compared to the ones for $T-$trees [6]).
Moreover, as the bit-map representation is a minimal representation for a trie, one may think that the compact-balanced trie approaches a theoretical optimum in terms of compactness.
The research presented in this paper could be extended in different ways: a compact representation for multidimensional data and a probabilistic analysis of the NIL-pointers, in both cases of random keys and of alphanumeric keys.

## References

[1] R. Bayer and K. Unterauer. Prefix *B*-trees. *A.C.M. Transactions on Database Systems*, 2(1):11–26, March 1977.

[2] H. Boral, W. Alexander, L. Clay, G. Copeland, S. Danforth, M. Franklin, B. Hart, M. Smith, and P. Valduriez. Prototyping Bubba, a highly parallel database system. *I.E.E.E. Transactions on Knowledge and Data Engineering*, 2(1), March 1990.

[3] W. de Jonge, A. S. Tenenbaum, and R. P. van de Riet. Two access methods using compact binary trees. *I.E.E.E. Transactions on Software Engineering*, 13(7):799–809, July 1987.

[4] T. Johnson and D. Shasha. Utilization of *B*-trees with inserts, deletes and modifies. In *Proc. of the 8th ACM SIGACT-SIGMOD Conference on Principles of Databases Systems*, pages 235–246, March 1989. Philadelphia.

[5] J. Kouacou-Kouadio. *Adressage d'Informations Structurées*. PhD thesis, Université Paris IX Dauphine, January 1986.

[6] T. J. Lehman and M. J. Carey. A study of index structures for main memory database management systems. In *Proc. of the 12th International VLDB Conference*, pages 294–303, June 1986. Kyoto.

[7] W. Litwin. Trie hashing. In *Proc. of ACM-SIGMOD Conference*, pages 12–29, April 1981. Ann Arbor, Michigan.

[8] W. Litwin and D. B. Lomet. A new method for fast data searches with keys. *I.E.E.E. Software*, pages 16–24, March 1987.

[9] P. Nicodème. Compact balanced tries. In J. van Leeuwen, editor, *Algorithms, Software, Architecture, Information Processing 92*, volume 1, pages 477–483. North-Holland, 1992. Proceedings of the IFIP 12th World Computer Congress, Madrid. Also available as INRIA Research Report n° 1533.

[10] R. Orlandic and J. Pfaltz. Compact 0-complete trees. In *Proc. 14th VLDB conference*, pages 372–381, August 1988. Los Angeles, California.

[11] R. Orlandic and J. Pfaltz. Analysis of compact 0-complete trees. In *Lecture Notes in Computer Science*, pages 362–371. Springer Verlag, August 1989. Szeged, Hungaria.

# 26

# Performances d'algorithmes de recherche de motifs sous divers modèles probabilistes

Mireille Régnier
INRIA, Rocquencourt

[résumé par Abalo Baya]

Le problème de recherche de motifs consiste à trouver toutes les occurrences d'un motif $p$ dans un texte $t$, ou la première occurrence, $p$ et $t$ étant des mots sur un alphabet $\mathcal{A}$ à $q$ éléments. La complexité des algorithmes de résolution d'un tel problème est habituellement, et ce sera le cas ici, le nombre de comparaisons de caractères entre le texte et le motif. Dans cet exposé l'auteur présente les résultats qu'il a établis sur la complexité en moyenne des algorithmes de Morris-Pratt [3] et de Boyer-Moore [1]. On considère un modèle de probabilité stationnaire, ce qui généralise les résultats déjà connus sous le modèle uniforme.

## 1 Algorithme de Morris-Pratt

Cette section décrit l'algorithme de Morris-Pratt (MP) et la plus utilisée de ses variantes, i.e. l'algorithme de Knuth-Morris-Pratt (KMP). Ces deux algorithmes lisent le texte $t$ de gauche à droite (la lecture de droite à gauche étant interdite) dans le but de trouver le motif $p$ recherché. La comparaison à réaliser à un instant donné est déterminée par un pointeur de texte $PT$ et un pointeur de motif $PM$ : Si $PT = i$ et $PM = j$, alors le $i^{\text{ème}}$ caractère du texte est comparé au $j^{\text{ème}}$ caractère du motif. Après une comparaison, ces pointeurs sont mis à jour en fonction du résultat de celle-ci. Si les deux caractères sont identiques (on dit qu'il y a *match*), alors on incrémente chaque pointeur d'une unité (sauf dans le cas où le motif est trouvé). En revanche, si ces deux caractères sont distincts (il y a *mismatch*) ou si $p$ est trouvé, la nouvelle comparaison est déterminée par le plus grand bord $p''$ du préfixe $p'$ de $p$ déjà trouvé, i.e. le plus grand sous-mot $p''$ tel que $p''$ soit à la fois un préfixe et un suffixe stricts de $p'$. On définit la fonction SUIVANT pour tout j par : $p'$ étant le préfixe de longueur $j$ de $p$, $p''$ son plus grand bord, de longueur $k-1$, SUIVANT($j$)=$k$. Cela veut dire qu'après un *mismatch* ou une occurrence de $p$, $PM = k$. Prenons par exemple $p = 01201201345$ et $t = 0301201201101201201345$. Après le premier *match*, $PM$ et $PT$ sont incrémentés d'une unité. Le *mismatch* entre $t[2] = 3$ et $p[2] = 1$ implique que $PM = 1$ et le *mismatch* suivant ($t[2] \neq p[1]$) implique que $PM = 1$ et $PT = 3$. On assiste alors à 8 *matches* suivis d'un *mismatch*, $t[11] \neq t[9]$. Le plus grand bord de $p' = 01201201$ est 01201 et $PM$ prend la valeur 6. Comme $p[6] \neq t[11]$ et 01 est le plus grand bord de 01201, $PM$ devient 3. Finalement $p[3] \neq t[11]$ implique $PM = 1$ et un *mismatch*. On incrémente $PT$ d'une unité et on trouve ensuite le motif.

On remarquera que cet algorithme ne tient pas compte de l'information sur le caractère du texte où a lieu un *mismatch*. En fait, quand SUIVANT[$j$]=$k$, la comparaison qui suit, $t[i]?p[k]$, donne lieu à un *mismatch* lorsque $p[j] = p[k]$. Ainsi, SUIVANT[$j$] peut être redéfini comme étant le plus grand $k$ tel que $p[1]...p[k-1] = p[j+1-k]...p[j-1]$ et $p[j] \neq p[k]$ et c'est précisément l'option choisie dans l'algorithme de Knuth-Morris-Pratt.

## 2 Algorithme de Boyer-Moore-Horspool

Les algorithmes du type Boyer-Moore lisent le texte de droite à gauche. Comme pour Morris-Pratt, le motif est décalé vers la droite lorsqu'il y a *mismatch* ou lorsque le motif a été trouvé. Ce décalage se fait suivant un *match* heuristique ou une occurrence heuristique. On s'intéresse ici à une simplification due à Horspool et basée sur l'occurrence heuristique. Si $m$ désigne la longueur du motif $p$, alors la table de décalage est formellement définie par

$$d(x) = \min\{s : s = m \text{ ou } (1 \leq s < m \text{ et } p[m-s] = x)\}.$$

Intuitivement, si $x$, en position $i$ sur le texte, est un caractère sur lequel il y a un *mismatch*, on fait glisser le motif $p$ vers la droite jusqu'à ce que le caractère $x$ de $p$ (s'il y en a) le plus à droite vienne coïncider sur la position $i$ du texte. Les positions du texte alignées avec le dernier caractère de $p$ après un décalage sont appelées *têtes*.

## 3 Modèle probabiliste

On donne ici les notations, définitions et hypothèses nécessaires à la recherche d'un motif donné dans un texte aléatoire.

**Définition 1** *Soit $p$ un motif de longueur $|p| = m$. On note $C_n(p)$, le nombre moyen de comparaisons faites par un algorithme de recherche de motifs sur un texte aléatoire de longueur $n$ et dont les caractères ont une distribution donnée. $C_n(m)$ désigne la valeur moyenne de $C_n(p)$ pour une distribution des caractères du motif donnée et lorsque $p$ parcourt l'ensemble des motifs de longueur $m$.*

**Définition 2** *On désigne par $\{p_a\}_{a\in A}$ (resp. $\{q_a\}_{a\in A}$), la distribution des caractères dans un motif (resp. dans un texte), c'est-à-dire la probabilité pour que toute position dans le motif ou dans le texte soit occupée par un caractère $a$ donné est $p_a$ (resp. $q_a$). $(\{p_a\},\{q_a\})_{a\in A}$ est dite distribution motif-texte.*

**Notation.** Pour $i \geq 1$, $\sigma_i = \sum_{a\in A}(p_a q_a)^i$, et pour $1 \leq i < j$, $s_{j,i} = \sum_{a\in A} p_a^j q_a^i$.

## 4 Performances des algorithmes KMP et MP

Avant d'énoncer les résultats obtenus, on pose la définition suivante :

**Définition 3** *Soit $F$ un polynôme ou une série entière sur l'ensemble $\{\sigma_i\}$. La troncature $F_m$ est le polynôme déduit de $F$ en ne considérant que les monômes $\prod \sigma_{\alpha_i}^{\beta_i}$ tels que $\sum_i \beta_i\alpha_i \leq m$.*

**Théorème 1** *On se donne une distribution motif-texte $(\{p_a\},\{q_a\})_{a\in A}$ et un motif de longueur $m$. Alors le coût amorti, i.e. le nombre moyen de comparaisons effectuées par caractère de texte, $\frac{C_n^{MP}(m)}{n}$, tend vers une constante $L_m^{MP}$ quand $n \to \infty$. Et on a:*

$$\begin{aligned} L_m^{MP} &\sim 1 + (\sigma_1 - \sigma_1^m) - F_m(\{\sigma_i\}), \\ \text{où} \qquad F(\{\sigma_i\}) &= \sum_{l\geq 1}\left(\frac{(\sigma_1 - 1)\sigma_{l+1}}{(1-\sigma_{l+1})(1-\sigma_l)} - \frac{\sigma_{l+2}}{1-\sigma_{l+1}}\right. \end{aligned}$$

Le terme d'erreur est $o(\sigma_1^5\sigma_3)$ et on a même l'égalité pour $m < 5$.

**Corollaire 1** *Pour $m$ assez grand, on a $L_m^{MP} \sim 1 + \sigma_1 - \sigma_1\sigma_2$. De plus,*

$$L_4^{MP} = 1 + \sigma_1 - \sigma_1\sigma_2 + \sigma_3 - \sigma_1^2\sigma_2 + \sigma_2^2 - \sigma_1^4 - \sigma_1\sigma_3 + \sigma_4$$

$$L_3^{MP} = 1 + \sigma_1 - \sigma_1^3 - \sigma_1\sigma_2 + \sigma_3 \qquad L_2^{MP} = 1 + \sigma_1 - \sigma_1^2.$$

**Théorème 2** *On fait les mêmes hypothèses que dans le théorème précédent. Alors, le nombre moyen de comparaisons $L_m^{KMP}$ faites par l'algorithme KMP satisfait $C_n^{KMP} \leq L_m^{MP}$ et*

$$|L_m^{KMP} - L_m^{MP}| \leq [F(\{p_iq_i\})[(s_{2,1} - \sigma_2) + F(\{p_i^2q_i^2\})\sigma_2(s_{2,1} - \sigma_2)]]_m$$

*où $F$ est la série définie dans le théorème précédent.*

**Théorème 3** *Pour le modèle uniforme, $L_m^{MP}$ vérifie*

$$L_m^{MP} \sim 1 + \frac{1}{q} - \frac{1}{q^m} + \frac{q-1}{q^2} G_{m-1}(\frac{1}{q^2})$$

*où*

$$G(x) = \frac{x}{1-x} \sum_l P(x^l).$$

*et où $P$ est la série génératrice (connue) des mots primitifs (i.e. les mots qui ne sont puissance d'aucun autre mot). Pour tout $m > 6$, l'approximation est d'ordre $1/q^{11}$ et c'est un coût exact pour les petites valeurs de $m$.*

## 5 Performances de l'algorithme de Boyer-Moore

L'analyse des performances pour l'algorithme de Boyer-Moore se fait en deux étapes : évaluation du nombre de "têtes", puis calcul du nombre de comparaisons.

**Théorème 4** *Pour un motif $p$ de longueur $m$, $H_p^l$ désigne la probabilité pour que le $l^{\text{ème}}$ caractère soit une tête. Alors $H_p^l$ converge vers une probabilité $H_p^\infty$, avec*

$$H_p^\infty = \frac{1}{E_p[\text{Shift}]} \equiv \frac{1}{\sum_{a \in A} q_a s_p(a)}$$

*où $s_p(a)$ est le décalage calculé lorsque $a$ est une tête dans le texte et où $E_p[\text{Shift}]$ est le décalage moyen sur l'ensemble des $q$ valeurs de l'alphabet.*

**Notation.** Soit $H(q,m) = E(H_p^\infty)$ la probabilité de trouver une tête en une position donnée du texte lorsque $p$ parcourt tous les motifs de longueur $m$ sur un alphabet de cardinal $q$. On note $Ph(q)$, la limite de $H(q,m), m \longmapsto \infty$.

**Remarque.** M. Régnier et R. Baeza-Yates ont établi la valeur exacte de $Ph(q)$ pour les petites valeurs de $q$ et une estimation pour les grandes valeurs. On donne ici les coûts moyens de l'algorithme de Boyer-Moore à l'aide de $H(q,m)$ et de $Ph(q)$.

**Théorème 5** *Soit $C_n(m)$ le nombre moyen de comparaisons texte-motif pour les textes aléatoires de longueur $n$ et les motifs de longueur $m$. Pour une distribution de texte pas trop irrégulière, on a :*

$$L_m^{BM} = \frac{C_n(m)}{n} = H(q,m)(1+\sigma_1+2\sigma_1^2),$$

*ou pour de grands motifs, $L_m^{BM} \sim Ph(q)(1+\sigma_1+2\sigma_1^2)$ et l'approximation est majorée par $\sigma_1^2(1/q(q+1)min\{q_a : a \in A\})^2$.*

**Remarque.** Tous les résultats ci-dessus ont été finalement étendus pour des dépendances markoviennes entre les données dans [6].

## Références

[1] R. Boyer and J. S. Moore. A fast string searching algorithm. *Communications of the ACM*, 20:762–772, 1977.

[2] G. H. Gonnet and R. Baeza-Yates. *Handbook of Algorithms and Data Structures: in Pascal and C.* Addison–Wesley, second edition, 1991.

[3] D. E. Knuth, J. Morris, and V. Pratt. Fast pattern matching in strings. *SIAM Journal on Computing*, 6:323–350, 1977.

[4] M. Régnier. Knuth-Morris-Pratt algorithm: an analysis. In *MFCS'89*, volume 379 of *Lecture Notes in Computer Science*, pages 431–444. Springer-Verlag, 1989. Proc. Mathematical Foundations of Computer Science 89, Porubka, Poland.

[5] M. Régnier. Performance of String Searching Algorithms under Various Probabilistic Models. Rapport de recherche 1565, Institut National de Recherche en Informatique et en Automatique, 1991.

[6] M. Régnier. A language approach to string searching evaluation. In *Proceedings of Combinatorial Pattern Matching '92, Tucson, Arizona*, pages 15–26. Springer-Verlag, 1992.

[7] R. Sedgewick. *Algorithms.* Addison-Wesley, Reading, Mass., second edition, 1988.

# 27

# Analyse des arbres suffixes par motif coulissant

Philippe Jacquet
INRIA, Rocquencourt

[résumé par Danièle Gardy]

Dans cet exposé, Ph. Jacquet présente un travail effectué en commun avec W. Szpankowski, consacré à un modèle de *trie* (arbre digital) où les clés, à la différence du modèle usuel, sont fortement corrélées [2, 1].

## 1 Rappels et définitions

### 1.1 Tries

Les *tries* sont habituellement construits sur des clés qui sont des suites infinies de bits 0 et 1, ou plus généralement des mots infinis sur un alphabet fini $A$. Notons une propriété intéressante pour la suite: *soit $\sigma$ le chemin conduisant de la racine à la feuille où est stockée la donnée $X_i$; $\sigma$ est le plus petit préfixe de $X_i$ qui ne soit préfixe d'aucune autre clé $X_j$.*
La plupart des études sur les tries ont été faites en supposant que les clés qu'ils contiennent sont indépendantes (voir par exemple les résultats rassemblés dans [4, p. 488]). Cependant, certains problèmes sur les mots conduisent à s'intéresser à des tries construits sur des clés fortement corrélées, obtenues par exemple en prenant les suffixes d'un même mot. On parle alors d'*arbre suffixe*. Ce type d'arbre apparaît aussi dans des problèmes liés à la compression de données, ainsi l'algorithme de Lempel et Ziv, qui est à la base de la commande *compress* en Unix.

### 1.2 Arbres suffixes

Soit un alphabet $A$, et soit $\omega$ un mot de $A^*$; les *suffixes* de $\omega$ sont obtenus en supprimant successivement les premières lettres de $\omega$. Par exemple, prenons $\omega = 01100111...$; les quatre premiers suffixes de $\omega$ sont $\omega$ lui-même, $1100111...$, $100111...$ et $00111...$ Un *arbre suffixe* de taille $n$ est défini par rapport à un mot $\omega$; c'est le trie formé sur les $n$ premiers suffixes de $\omega$.

### 1.3 Modèle probabiliste

Les clés sont construites sur un alphabet $A$ de taille $V$, dont les lettres n'ont pas nécessairement toutes la même probabilité. Un mot $\omega$ est un élément de $A^*$, dont les lettres sont indépendantes (modèle de Bernoulli). Sur ce mot $\omega$, on construit alors un arbre suffixe en prenant les $n$ premiers suffixes.

## 2 Résultats

Il y a une abondante littérature sur les tries; on pourra se reporter entre autres à l'ouvrage [3] pour une synthèse récente. Il est donc naturel de chercher à s'y rattacher, et à utiliser des résultats déjà connus. L'approche de Ph. Jacquet et W. Szpankowski consiste justement à montrer la proximité de leur modèle avec le modèle classique d'un trie construit sur des clés indépendantes, en termes de fonctions génératrices. Leur résultat fondamental est le suivant:

$$E_{\text{suffixe}}[u^{|\text{chemin}|}] = E_{\text{trie}}[u^{|\text{chemin}|}] + O(n^{-\epsilon}).$$

Ils en déduisent la distribution de la hauteur d'une feuille de l'arbre, i.e. la longueur d'un suffixe d'un mot aléatoire $\omega$: soit $D_n$ la longueur moyenne d'un chemin vers un suffixe quelconque dans l'arbre suffixe d'ordre $n$ construit sur un mot aléatoire $\omega$, et posons $h_1 = -\sum_i p_i \log p_i$ et $h_2 = \sum_i p_i (\log p_i)^2$. Alors

$$E[D_n] = \frac{1}{h_1}(\log n + \gamma + \frac{h_2}{2h_1}) + P_1(\log n) + O(n^{-\epsilon}).$$

De plus, il est possible d'obtenir la variance et la loi limite de $D_n$, qui diffèrent suivant que les lettres de l'alphabet $A$ sont équiprobables ou non:

- Dans le cas où les lettres de $A$ n'ont pas toutes la même probabilité, la variance est d'ordre $\log n$:
$$var(D_n) = \frac{h_2 - h_1^2}{h_1^3} \log n + C + P_2(\log n) + O(n^{-\epsilon}).$$
De plus, la distribution de $D_n$ est asymptotiquement gaussienne.
- Dans le cas symétrique, $p_i = 1/V$ pour tout $i$, donc $h_2 = h_1^2$, et la variance tend alors vers une limite finie non nulle:
$$var(D_n) = \frac{\pi^2}{6(\log V)^2} + \frac{1}{12} + P_3(\log n) + O(n^{-\epsilon}).$$
Il existe là aussi une loi limite pour $D_n$, qui est ici de type double exponentielle:
$$\text{Proba}\{D_n \leq \log_V n + x\} \to exp(-V^{-x}).$$

## 3 Preuve du théorème fondamental

Soit $\mathcal{C}$ un ensemble de $n$ clés (corrélées ou non): $\mathcal{C} = \{X_1, X_2, \ldots, X_n\}$.

**Définition 1** *Soit $\sigma$ un mot fini; $\langle\sigma\rangle_{\mathcal{C}}$ est l'ensemble des indices $i$ tels que $\sigma$ soit un préfixe de la clé $X_i$ de $\mathcal{C}$.*

Pour un trie ou un arbre suffixe construit sur les clés d'un ensemble $\mathcal{C}$, notons $D_n^{\mathcal{C}}[i]$ la longueur du chemin allant de la racine vers la feuille contenant la clé $X_i$. On a la propriété suivante:

*$D_n^{\mathcal{C}}[i] \leq k$ si et seulement s'il existe une chaîne $\sigma$ de longueur $k$ qui n'est préfixe que de $X_i$, i.e. telle que $\langle\sigma\rangle_{\mathcal{C}} = \{i\}$.*

Les événements Proba($\langle\sigma\rangle_{\mathcal{C}} = \{i\}$) étant disjoints, on a:

$$\text{Proba}(D_n^{\mathcal{C}}[i] \le k) = \sum_{|\sigma|=k} \text{Proba}(\langle\sigma\rangle_{\mathcal{C}} = \{i\}).$$

Soit $D_n^{\mathcal{C}}$ la longueur moyenne du chemin de la racine jusqu'à une clé aléatoire de l'ensemble $\mathcal{C}$: $D_n^{\mathcal{C}}$ prend la valeur $D_n^{\mathcal{C}}[i]$ avec une probabilité $1/n$, quel que soit $i$. On montre (en conditionnant par rapport à l'indice $i$ retenu) que

$$\text{Proba}(D_n^{\mathcal{C}} \le k) = (1/n) \sum_{i=1}^{n} \text{Proba}(D_n^{\mathcal{C}}[i] \le k).$$

Posons $D_n(u) = E[u^{D_n}]$; on a $D_n(u) = \sum_{\mathcal{C}} \text{Proba}(\mathcal{C}) E[u^{D_n^{\mathcal{C}}}]$. En utilisant l'abréviation

$$f(\sigma, i) = \sum_{\mathcal{C}} \text{Proba}(\mathcal{C})\ \text{Proba}(\langle\sigma\rangle_{\mathcal{C}} = \{i\}),$$

où la somme est prise sur tous les ensembles de clés $\mathcal{C}$ admissibles pour le problème étudié, on trouve que

$$D_n(u) = \frac{1}{n}(1-u) \sum_{\sigma} u^{|\sigma|} \sum_{i=1}^{n} f(\sigma, i),$$

et l'étape suivante est de calculer $\sum_i f(\sigma, i)$. Soit $\mathcal{L}$ un sous-ensemble de $\{1, 2, \ldots, n\}$, et soit

$$P(\mathcal{L}, \sigma) = \sum_{\mathcal{C}} \text{Proba}(\mathcal{C})\ \text{Proba}(\mathcal{L} \subset \langle\sigma\rangle_{\mathcal{C}}).$$

Par un argument d'inclusion-exclusion, on montre que, pour tout ensemble de clés $\mathcal{C}$,

$$\text{Proba}(\langle\sigma\rangle_{\mathcal{C}} = \{i\}) = \sum_{j=1}^{n} (-1)^{j+1} \sum_{|\mathcal{L}|=j,\ i\in\mathcal{L}} \text{Proba}(\mathcal{L} \subset \langle\sigma\rangle_{\mathcal{C}}).$$

En sommant sur les ensembles $\mathcal{C}$, on obtient

$$f(\sigma, i) = \sum_{j=1}^{n} (-1)^{j+1} \sum_{|\mathcal{L}|=j,\ i\in\mathcal{L}} P(\mathcal{L}, \sigma).$$

Pour obtenir $D_n(u)$, on peut alors calculer $P(\mathcal{L}, \sigma)$, puis $f(\sigma, i)$, ce qui va être fait ci-dessous d'abord pour des tries construits sur des clés indépendantes, puis pour des arbres suffixes.

### 3.1 Tries sur des clés indépendantes

Supposons que $\sigma = a_1 a_2 \cdots a_k$, et notons $p(\sigma) = p_{a_1} p_{a_2} \ldots p_{a_k}$. On a alors $P(\mathcal{L}, \sigma) = p(\sigma)^{|\mathcal{L}|}$ et donc $f(\sigma, i) = p(\sigma)(1 - p(\sigma))^{n-1}$. On en tire:

$$D_n(u) = \frac{1}{n}(1-u) \sum_{\sigma} n\ p(\sigma)(1-p(\sigma))^{n-1} u^{|\sigma|}.$$

Définissons $D_T(z, u) = \sum_n n D_n(u) z^n$:

$$D_T(z, u) = (1-u) \sum_{\sigma} u^{|\sigma|} \frac{p(\sigma) z}{(1 - z + p(\sigma) z)^2}. \tag{1}$$

### 3.2 Arbres suffixes

Par un raisonnement analogue, bien que plus complexe, en posant $D_S(z,u) = \sum_n n D_n(u) z^n$, où cette fois $D_n(u) = E(u^{D_n})$ est calculé sur les arbres suffixes, on obtient:

$$D_S(z,u) = (1-u)\sum_\sigma \frac{p(\sigma)z}{\left((1-z)(1+a_\sigma(z)) + p(\sigma)z^{|\sigma|}\right)^2} u^{|\sigma|}, \tag{2}$$

où $a_\sigma$ est le polynôme d'auto-corrélation de $\sigma$, défini par la somme suivante, avec $\sigma'$ décrivant l'ensemble des préfixes de $\sigma$ qui sont aussi suffixes de $\sigma$ (et bien sûr $\sigma' \neq \sigma$):

$$a_\sigma(z) = \sum_{\sigma'} \frac{p(\sigma)}{p(\sigma')} z^{|\sigma|-|\sigma'|}.$$

Par exemple [1, p. 16], soit $\sigma = abaaba$ de longueur 6. Alors l'ensemble des mots qui sont à la fois préfixes et suffixes de $\sigma$ est $\mathcal{E} = \{a, aba\}$, et

$$a_\sigma(z) = \sum_{\sigma' \in \mathcal{E}} \frac{p(\sigma)}{p(\sigma')} z^{|\sigma|-|\sigma'|} = p_a^2 p_b z^3 + p_a^3 p_b^2 z^5.$$

L'idée de la démonstration conduisant à la formule (2) est la suivante: lorsque les clés sont des suffixes d'un même texte (supposé infini), et lorsque ces suffixes débutent à des positions distantes d'au moins $k$, $k$ étant la longueur de $\sigma$, alors ces suffixes peuvent être considérés comme indépendants. On regroupe alors les premières positions, jusqu'à ce qu'on ait un trou de $k$; les suffixes suivants sont traités indépendamment des précédents.

### 3.3 Comparaison des deux fonctions

A défaut de pouvoir étudier directement la fonction définie par l'équation (2), on va la comparer avec la fonction analogue pour les tries, définie par l'équation (1), et qui a été bien étudiée. Cela conduit à s'intéresser à la différence

$$Q_n(u) = \frac{1}{1-u}(D_n^{\text{trie}}(u) - D_n^{\text{suffixe}}(u)).$$

D'après la formule de Cauchy, $Q_n(u)$ s'exprime par une intégrale:

$$Q_n(u) = \frac{1}{2i\pi n(1-u)} \oint (D_T(z,u) - D_S(z,u)) \frac{dz}{z^{n+1}}.$$

Ici intervient, pour chaque mot fini $\sigma$, la racine de plus petit module $A_\sigma$ du polynôme $R_\sigma(z) = (1-z)(1+a_\sigma(z)) + p(\sigma)z^{|\sigma|}$. Notons $C_\sigma = R'_\sigma(A_\sigma)$ et $D_\sigma = R''_\sigma(A_\sigma)$. On montre que, pour un $B$ "convenable":

$$Q_n(u) = \frac{1}{n}\sum_\sigma u^{|\sigma|} p(\sigma) \left( A_\sigma^{-n}\left(\frac{n}{A_\sigma C_\sigma^2} + \frac{D_\sigma}{C_\sigma^3}\right) - n(1-p(\sigma))^{n-1} \right) + O(B^{-n}),$$

puis que

$$Q_n(u) = g_n(u) + O(\frac{1}{n}),$$

avec

$$g_n(u) = \sum_{\sigma} u^{|\sigma|} p(\sigma) f_\sigma(n)$$

et avec

$$f_\sigma(x) = \frac{A_\sigma^{-x} - e^{-x}}{A_\sigma C_\sigma^2} - \frac{(1-p(\sigma))^x - e^{-x}}{1-p(\sigma)}.$$

Il reste alors à voir qu'il existe $\epsilon > 0$ tel que $g_n(u)$ est $O(n^{-\epsilon})$. Pour cela, on utilise des propriétés qui relient le comportement asymptotique d'une fonction (ici $g_n(u)$, en tant que fonction de $n$) à la bande fondamentale de sa transformée de Mellin, i.e. à l'ensemble sur lequel cette transformée est "naturellement" définie. On calcule donc, pour chaque $\sigma$, la transformée de Mellin de $f_\sigma$:

$$f_\sigma^*(s) = \int_0^{+\infty} f_\sigma(x) x^{s-1} ds = \Gamma(s) \left( \frac{(\log A_\sigma)^{-s} - 1}{A_\sigma C_\sigma^2} - \frac{(-\log(1-p(\sigma))^{-s} - 1}{1-p(\sigma)} \right).$$

Pour un motif $\sigma$ donné, cette transformée est définie sur $]-1, +\infty[$. Lorsqu'on somme sur tous les mots $\sigma$, on peut montrer que, pour un certain $\epsilon > 0$, $g^*(s) = \sum_\sigma u^{|\sigma|} p(\sigma) f_\sigma^*(s)$ reste définie sur $]-1, \epsilon[$. D'où la majoration cherchée: $g_n(u) = O(n^{-\epsilon})$, à quelques détails techniques près.

## Références

[1] Ph. Jacquet and W. Szpankowski. Autocorrelation on words and its application: Analysis of suffix trees by string-ruler approach. Technical Report 1106, Institut National de Recherche en Informatique et en Automatique, October 1991. Accepted for publication in Journal of Combinatorial Theory, Series A.

[2] Ph. Jacquet and W. Szpankowski. What can we learn about suffix trees from independent tries? In *Second Workshop on Algorithms and Data Structures*, Lecture Notes in Computer Science, pages 228–239, Ottawa (Canada), August 1991. Springer Verlag.

[3] H. M. Mahmoud. *Evolution of Random Search Trees*. John Wiley, 1992.

[4] J. Vitter and Ph. Flajolet. Analysis of Algorithms and Data Structures. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume A: Algorithms and Complexity, chapter 9, pages 431–524. North Holland, 1990.

# 28

# Fast Two Dimensional Pattern Matching

Mireille Régnier
INRIA, Rocquencourt

[summary by Pierre Nicodème]

**Abstract**

We address the problem of finding a $m \times m$ pattern in a $n \times n$ text. We conjecture that, with a constant (i.e. $O(m^2)$) additional memory, this complexity lies between $[1+O(\frac{1}{m})]n^2$ and $[2+O(\frac{1}{m})]n^2$, which is different from the linear result in dimension 1. We provide an algorithm that achieves this goal with good average performance, that can be easily coded and that can be made alphabet independent.

## 1 Introduction and State of the Art

First algorithms for 2D pattern matching [4, 5, 9, 15] have been rather rough extensions to two dimensions of 1D pattern matching paradigms, that did not really used the specificity of 2D. Hence, all had an average case of $n^2$ at least. Additionally, most would need a $O(n)$ extra-space. The only exception is [15], but it needs a $O(n^2)$ extra-space. By dividing the text into subpieces, [9] reduces its extra-space to $O(m^2)$, but the price to pay is a substantial increasing of the average number of comparisons. Moreover, all are alphabet dependent.
Such performance sounds very poor when compared to 1D theoretical and practical results. For the average case, a $n\frac{\log m}{m}$ theoretical bound has been proved in [14], and it is achieved, for uniform distributions, by a rough algorithm in [10]. Boyer-Moore and its variants are sublinear for "non pathological" distributions and non-binary alphabets [3, 13]. Worst-case complexity relies between 1 and 2 for most practical algorithms [6], and theoretical worst case complexity was recently proved to be $1+O(\frac{1}{m})$ [7].
In [2], 2D specificity is used, and a first sublinear algorithm in the average is proposed. It drastically improves on previous ones, as average performance becomes $O(\frac{n^2}{m})$ with $O(m^2)$ extra-space. Additionally, the worst case does not depend on the pattern size, being $O(n^2)$, and it runs on-line.

## 2 The algorithm

### 2.1 Formalism

Worst-case complexity is related to the *maximal number of occurrences* of a searched pattern $P$ in all possible texts, hence, to maximal coverings of the text by the pattern. As $P$ can overlap with himself, these may not be tilings. This can be expressed as a function of a canonical decomposition of $P$. Let us say a few words on 1D complexity [8]. It is proved in [11] that a self-overlapping pattern can be written $p = u(vu)^m, m \geq 1, u \neq \epsilon$, where $vu$ is *primitive*, i.e. not the power of any

word $w$. The pattern $p$ is *periodic* if and only if $m \geq 2$. As discussed in [12], the decomposition is not unique although at most one satisfies $m \geq 2$. Nevertheless, two coverings $t = u(vu)^*$ and $t = z(wz)^*$ cannot be interleaved. Hence, 1D worst case complexity is linear [7].
This property disappears in 2D pattern matching. Two coverings can be interleaved.

**Definition 1** *An alignment of a pattern $P$ with himself may be characterised by canonical coordinates: a 3-tuple $(x, y, dir)$. One of the two occurrences, say $P_r$, has a left corner inside the area defined by the second one, $P_l$; $x$ and $y$ are the row and column indices so defined, and $dir$ is a boolean set to true (resp. false) when this is an upper (resp. lower ) corner.*

We will refer to *dir* as the direction of the alignment, and use the terminology of *down* or *up* alignment.

**Definition 2** *An alignment of a pattern $P$ with himself is* **consistent** *if no mismatch occurs in the overlapping area.*

One also says that $P$ self overlaps. When the overlapping area includes the center(s), $P$ is periodic. We define here the repetitions in the pattern that will allow to speed up the searching process (compared to the naive search). We also define an order:

**Definition 3** *Given two overlapping potential matching areas PM and PM' with canonical coordinates relative to the beginning of the text $(x, y)$ and $(x', y')$, we define the order:*

$$PM \preceq PM' \Longleftrightarrow y < y' \text{ or } (\ y = y' \ and \ x \leq x'\ )\ .$$

**Definition 4** *Two potential matching areas are said to be (fully) consistent in either one of the two cases:*

* *they do not overlap,*
* *the associated alignment is consistent.*

**Definition 5** *A set of overlapping areas is said fully consistent if its elements are mutually consistent. Its left border is the column coordinate of its minimal elements.*

Remark that all column coordinates of an overlapping area range between the left border $l$ and $l + 2m$. We also define a *canonical checking order* for all patterns in the text: say from left to right and top to bottom. This yields an interesting property that we will use:

**Lemma 1** *Let $PM'$ be a potential matching area. We define property $(P1)$ by:*

$(P1)$*: All checked positions are*

* *either matching positions of some smaller consistent overlapping potential matching areas.*
* *or mismatching positions of non consistent potential matching areas.*

Then, we will use an array MISMATCH$(x, y, dir)$ of couples of integers $(X,Y)$: $(X,Y)$ is either $(0,0)$ for a consistent alignment or the coordinates of the first mismatching position. This allows checking consistency in constant time. Additionally, this provides a criterium to discard a potential alignment of $P$ with some potential matching area $PM'$: namely, whenever some $PM \preceq PM'$ has been checked until mismatching position $\alpha$ interior to the overlapping area. This criterium is checked in *constant time* (procedure *Check* in our code). Remark that the preprocessing is trivially achieved by checking $(\sum_1^m i)^2$ characters, which is $O(m^4)$.
We are now ready to explain our algorithm. It relies on a division of the whole text in *slabs* of height $m$, delimited by rows $km$, $k \in \mathbb{N}$. These rows are said *primary* rows, and the characters in them *primary* characters (versus *secondary* rows and characters). Then, any potential matching area will intersect one primary row, and only one. This intersection will be any row $p_i$, $i = 1 \ldots m$ of the pattern. Hence, we proceed in two stages. First, a multi-string searching of strings $p_i$ is performed on primary rows, defining a potential matching area. Any multi-string searching automaton [1] can be used. We name this automaton "automaton A", this stage *horizontal* or *primary* search. The second stage checks secondary characters. So far, this "slab method" does not significantly differ from [2] that provided a good average case: $O(\frac{n^2}{m})$. In order to ensure simultaneously an optimal worst case (or close to optimal) we delay secondary search until we know "enough information" to the right. And while performing it, we make use of all known information to the left. Our horizontal search is completed as follows: initialisation runs automaton A until some pattern is found, uniquely defining a potential matching area, *First*, with column coordinate *col*. The stationary stage consists of both checking the consistency of potential matching areas, and restarting the automaton A (which remains loaded), whenever the leftmost potential matching areas have been successfully checked or discarded. In the stationary stage, we run $A$ to detect the next PM in the range $col, col + m - 1$. *ListCandidate* is maximal if and only if no more exists. Else, we check PM against every candidate from *ListCandidate*. If it is consistent with all of them, it is inserted to it. If not, a "Duel Strategy" using *Check* allows to kill a (strict) subset of *ListCandidate* $\cup \{PM\}$. Remark that whenever *First* is killed in this process, the left border shifts to the right and automaton $A$ is restarted. Then, we start the effective secondary check, checking its minimal element, *First*, and avoiding to redo comparisons on positions already checked during the same effective secondary checking; this is done by maintaining a list of 2m-1 intervals of characters already checked. When it ends, we update *ListCandidate* using our *Check* procedure if last comparison was a mismatch and restart the horizontal search. Remark that all elements remaining in *ListCandidate* being consistent with the last checking, property $(P1)$ is still ensured. For a sake of clarity of the description and of the code, we assume now that all $p_i$ are different. We delay after the discussion of worst case complexity the description of the general case.
Finally, we show how to avoid multiple comparisons in secondary search.

**Lemma 2** *Let $PM$ be a valid candidate, i.e. a potential matching area for which* Secondary Search *is called, and col its column coordinate. For any secondary row $i, 1 \leq |i| \leq m-1$, the largest column index of text positions scanned in a secondary search, $j$, satisfies:*

* *either $j < col$,*
* *or all positions between col and $j$ are matching positions.*

**Proof.** From our *ListCandidate* construction, whenever *SecondarySearch* is called for two overlapping areas, they are consistent. Assume now $j \geq col$, and let $PM_0$ be the area for which $t[i, j]$ was

scanned. If $t[i,j]$ was a mismatch, then $PM$ would have been discarded from *ListCandidate* after this *SecondarySearch* call. Then, it was a match for $PM_0$, hence for $PM$. □

```
SlabSearch( (text, n × n), (pat, m × m) )
for( k ← m; k ≤ n; k ← k + m ) {                    /* k: current primary row index */
    q ← q0;                                          /* searching a fully consistent set */
    for( j ← 1; j ≤ n; j ← j + 1 ) {
        Repeat { q ← A(q, text[k, j]); i ← Output(q); } until (i ≠ 0);      /* i: found string index */
        Insert(ListCandidate,(i, j));
        while (ListCandidate ≠ ∅)
        {
          while (j < col + m & j ≤ n)                /* col: column index of First */
          {
            q ← A(q, text[k, j]); i ← Output(q);
            if i ≠ 0 then {
              for (i', j') ∈ ListCandidate {          /* Check (i, j) consistency with ListCandidate */
                  Survive ← true;
                  (X, Y) ← MISMATCH(|i' − i|, j' − j, sign(i' − i)) ;
                  if X + Y ≠ 0 then                                  /* kill one candidate */
                    if (Survive ← ¬(text([k − i' + X, j' + Y] == Pat[X, Y])) then
                      Delete(ListCandidate,(i', j'))     /* Mismatch on (i', j') in ListCandidate */
                      else break;   /* Mismatch on (i, j): stop checking ListCandidate, restart A */
                  };                                                  /* end for */
              if Survive then Insert(ListCandidate,(i, j));        /* (i, j) consistency */
              };                                                        /* endif */
            j ← j + 1;
          }                                                          /* end second while */
        SecondaryCheck(First);
        Update(ListCandidate);
        }                                  /* Secondary search end; restart ListCandidate search */
    }
}
```

## 3 2D-Complexity

We defer the precise study of 2D-complexity. This study will make an exhaustive analysis of plane covering and tiling by patterns not overlapping, or overlapping in different ways. Theoretical bounds will have to be derived from this analysis.

## 4 Alphabet Independency

As we said above, our algorithm depends on the alphabet only by the multistring searching automaton (and the preprocessing). It can be made alphabet independent in the following way [2]; choose any right to left automaton, represent it as a $m$-ary tree and map it to a binary tree in the classical way. A final state is attained in $(l + r)$ transitions, where $l$ and $r$ are the number of left and right branchings on the path to this state. Note that a left branching is associated to a match while a right one is a shift to another string. Consider now a step where one stopped on $(l_0, r_0)$. The rough upper bound $l_0 + r_0 \leq 2m$ holds. Assume first the (strictly positive) shift $s$

also is greater than 2. If no string was found, then 2 columns are definitely discarded, hence a cost $\frac{2m}{(m-1)s} \leq 1 + O(\frac{1}{m})$. This bound also holds when $p_i$ is found but *SecondarySearch* not called later on the alignment so defined. Now, assume $p_i$ is found and defines an alignment on which *SecondarySearch* is called later. Within a range of $m$, we have $k$ steps with a cost $\sum_{i\geq 1}(l_i + r_i)$. By construction $\sum l_i \leq m$ and $\sum r_i \leq m + m$ (number of strings and potential number of fails). This finally yields a $4m$ bound, and a $O(1)$ amortised cost for *primary* characters. Last, we consider a shift $s = 1$ and assume a strict suffix *Suf* of some string is found. If $|Suf| \leq m - 2$, we reason as above. Else, we have $r_0 + r_1 \leq m$, unless $Suf = a^*$. More generally, we either have $\sum r_i \leq m$ within a range of $m$ or we are led to previous cases. The end of the proof for $p_i = a^{m-1}*$ is similar and we skip the technical details.
Hence, the difference to linear cost comes from slabs overlaps on secondary rows.

# References


[1] A. V. Aho and M. Corasick. Efficient String Matching. *Communications of the ACM*, 18(6):333–340, 1975.

[2] R. Baeza-Yates and M. Régnier. Fast algorithms for two dimensional and multiple pattern matching. In *SWAT'90*, volume 447 of *Lecture Notes in Computer Science*, pages 332–347. Springer-Verlag, 1990. Proc. Swedish Workshop on Algorithm Theory, Bergen, Norway. To appear in IPL.

[3] R. Baeza-Yates and M. Régnier. Average running time of Boyer-Moore-Horspool algorithm. *Theoretical Computer Science*, 92:19–31, 1992.

[4] T. Baker. A technique for extending rapid exact string matching to arrays of more than one dimension. *SIAM Journal on Computing*, 7:533–541, 1978.

[5] R. Bird. Two dimensional pattern matching. *Information Processing Letters*, 6:168–170, 1977.

[6] R. Cole. Tight Bounds on the Complexity of the Boyer-Moore string matching algorithms. In *SODA'91*, pages 224–233. SIAM, 1991. Proc. 2-nd SIAM-ACM Symp. on Discrete Algorithms, San Francisco, USA.

[7] R. Cole and *al.* 1D pattern matching complexity, 1992. Preprint.

[8] L. Colussi, Z. Galil, and R. Giancarlo. On the exact Complexity of string matching. In *FOCS'90*, pages 135–143. IEEE, 1990. Proc. 31-st Annual IEEE Symposium on the Foundations of Computer Science.

[9] R. Karp and M. Rabin. Efficient randomized pattern-matching algorithms. *IBM Journal of Research and Development*, 31:249–260, 1987.

[10] D. E. Knuth, J. Morris, and V. Pratt. Fast pattern matching in strings. *SIAM Journal on Computing*, 6:323–350, 1977.

[11] M. Lothaire. *Combinatorics on Words*. Addison-Wesley, Reading, Mass., 1983.

[12] M. Régnier. Knuth-Morris-Pratt algorithm: an analysis. In *MFCS'89*, volume 379 of *Lecture Notes in Computer Science*, pages 431–444. Springer-Verlag, 1989. Proc. Mathematical Foundations of Computer Science 89, Porubka, Poland.

[13] M. Régnier. A language approach to string searching evaluation. In *Proceedings of Combinatorial Pattern Matching '92, Tucson, Arizona*, pages 15–26. Springer-Verlag, 1992.

[14] A. C. Yao. The complexity of pattern matching for a random string. *SIAM Journal on Computing*, 8:368–387, 1979.

[15] R. F. Zhu and T. Takaoka. A technique for two-dimensional pattern matching. *Communications of the ACM*, 32(9):1110–1120, 1989.

# Part V

# Computational Number Theory

# 29

# Histoire et application des machines de crible numérique

Hugh C. Williams
University of Manitoba, Winnipeg

[résumé par Abalo Baya]

Une machine est dite *machine crible* si elle permet de résoudre un ou plusieurs systèmes de congruences linéaires à une variable. Le mécanisme de résolution de tels systèmes est la recherche exhaustive sur un ensemble d'entiers fixé. Une telle approche peut paraître naïve, mais pour certains problèmes, on ne connaît pas de méthode plus efficace. Dans cet exposé l'auteur fait l'historique sur les machines cribles et montre comment elles ont été utilisées pour obtenir des informations portant sur divers problèmes relatifs à la théorie des nombres.

Intéressons-nous d'abord à l'un des problèmes fondamentaux de l'exposé. On se donne

1. un intervalle $[A, B]$ avec $B > A$,
2. $k$ entiers $m_1, m_2, \ldots, m_k$ premiers entre eux ($m_i > 1, i = 1, 2, \ldots, k$) appelés les "modulos",
3. $k$ ensembles de résidus

$$R_i = \{r_{ij} \mid 0 \leq r_{ij} < m_i\},\ i = 1,\ 2,\ \ldots, k.$$

Le problème consiste à trouver tous les $x$ tels que $A \leq x < B$ et $x \bmod m_i \in R_i$, $i = 1, 2, \ldots, k$. Certains cas particuliers classiques de ce problème peuvent être résolus à l'aide d'un algorithme efficace (c'est par exemple le cas du problème des restes chinois qui se résout à l'aide de l'algorithme d'Euclide), alors que pour les autres on ne connaît pas de méthode plus efficace que la résolution par des machines cribles. Les premières machines cribles de résolution d'une équation diophantienne par la méthode d'exclusion (Gérardin, Kraitchic, P. & E. Carissan (1912)) sont restées à l'état de prototype. La machine crible de Carissan (1919) est à commande manuelle et utilise 14 modulos dans la méthode d'exclusion : elle trie 35 à 40 nombres par seconde. Quant au crible optique de Lehmer (1932), il atteint une performance de 5000 tris par seconde. Par ailleurs, Lehmer a construit une machine automatique pouvant résoudre certains problèmes de crible et cette méthode a permis de factoriser des grands entiers tels que $(2^{136} + 1)/98564897$. Jusqu'à 1970, cette méthode était la plus efficace connue pour la factorisation des entiers. Le tableau suivant donne pour chaque machine, l'année de sa réalisation, le nombre de modulos utilisés et sa performance en nombres de tris par seconde.

| Machine | année | nb. modulos | nb. tris/s |
|---|---|---|---|
| E. Carissan | 1919 | 14 | $35-40$ |
| "Bicycle Chain" | 1926 | 19 | 50 |
| "Optical Gears" | 1932 | 30 | 5000 |
| "16 mm Movie Film" | 1936 | 18 | 50 |
| A. Gérardin | 1937 | ? | ? |
| SWAC | 195x | ? | 1450 |
| IBM7094 | 196x | 21 ou 22 | 100000 |
| DLS-127 | 1966 | 31 | 1000 000 |
| DLS-157 | 1969(?) | 37 | 1000 000 |
| ILLIAC IV | 196x | 64 | 15 000 000 |
| Registre à Décalage | 1975 | 42 | 20 000 000 |
| UMSU | 1983 | 32 | 133 000 000 |
| SSU | 1991 | 30 | 200 000 000 |

Comme application, ces machines ont été utilisées pour le calcul des polynômes quadratiques dont les valeurs comportent une forte densité de nombres premiers, pour la recherche de la solution du problème des pseudo-carrés et du problème d'Erdős. Dans la dernière partie de l'exposé, l'auteur présente un dispositif de crible qu'il a mis au point pour la recherche du plus grand pseudo-carré. La performance d'un tel dispositif est de $8.87 \times 10^{11}$ tris par seconde.

## Références

[1] E. Carissan. *London Math. Soc. Lecture Notes*, 154:38–75.

# 30

# Probabilistic Primality Testing

A. Oliver L. Atkin
University of Illinois, Chicago

[summary by François Morain]

**Abstract**

The aim of this talk is to give a strong probabilistic pseudoprimality test that recognises a maximal number of composite numbers as fast as possible. As a by-product, it is shown how to get "free" square-roots of certain elements of $\mathbb{Z}/p\mathbb{Z}$.

## 1 Introduction

The prototype of pseudoprime tests is Fermat's theorem: If $p$ is a prime and $a$ an integer prime to $p$, then

$$a^{p-1} \equiv 1 \bmod p.$$

A pseudoprime to base $a$ (psp-$a$) is a composite number $N$ such that

$$a^{N-1} \equiv 1 \bmod N.$$

For all $a$, there exists an infinite number of psp-$a$. Moreover, there are numbers $N$ such that $N$ is a psp-$a$ for all $a$. (These numbers are called Carmichael numbers.) A refinement of this test consists in writing $N = 1 + N_0 2^t$ with $N_0$ odd and

$$a^{N-1} - 1 = (a^{N_0} - 1)(a^{N_0} + 1)(a^{2N_0} + 1) \cdots (a^{2^{t-1}N_0} + 1).$$

If $N$ is prime, it divides the left-hand side and so must divide one of the numbers on the right hand side. If $N$ is composite and divides one of the numbers on the right, then $N$ is called a strong pseudoprime to base $a$ (spsp-$a$). As for psp-$a$'s, there is an infinite number of spsp-$a$.
A classical way of proving the primality of $N$ is to test whether $N$ is spsp-2 (say) and then rely on some more sophisticated algorithm to finish the proof [3, 1]. In certain cases, however, one might want to be as confident as possible that $N$ is prime, without using the above-mentioned methods. Typically, one wants a test whose running time is at most five times that of a modular exponentiation with the lowest error probability possible.

## 2 $q$-strong pseudoprimes

One way of achieving this is to find small factors of $N-1$:

$$N-1 = q^t N_0$$

with $q$ a "small" prime and $N_0$ prime to $q$. For a given $a$, put

$$b \equiv a^{N_0} \bmod N.$$

If $b \equiv 1 \bmod N$, one chooses another $a$ and try again. Otherwise, there exists a value of $i$ such that

$$b^{q^{i-1}} \not\equiv 1 \bmod N$$

but

$$b^{q^i} \equiv 1 \bmod N.$$

Put $B = b^{q^{i-1}}$. If $N$ is prime, then

$$N \mid B^q - 1 = (B-1)(B^{q-1} + B^{q-2} + \cdots + 1)$$

and therefore

$$N \mid B^{q-1} + B^{q-2} + \cdots + 1.$$

If $N$ is composite and the preceding relation is true, then $N$ is called a $q$-strong pseudoprime to base $a$ ($\mathrm{spsp}_q(a)$). In that case, $B$ behaves like a $q$-th root of unity modulo $N$. We shall use this fact in section 4.

## 3 Lucas sequences

Let $A$ be a small rational (i.e., $A = u/v$ with $u$ and $v$ small) such that $\Delta = A^2 - 4$ is a quadratic non-residue modulo $N$. Let $\alpha$ and $\beta$ be the two distinct roots of

$$X^2 - AX + 1 = 0$$

and put $S_n = \alpha^n + \beta^n = \alpha^n + \alpha^{-n}$. (Note that computing $S_n$ can be done in $O(\log n)$ steps using the relations

$$S_{2n} = S_n^2 - 2 \bmod N,\ S_{2n+1} = \frac{S_{2n+2} + S_{2n}}{A} \bmod N.)$$

If $N$ is a prime, and since $\Delta$ is a quadratic non-residue, $\alpha$ is in the Galois field $F = \mathrm{GF}(N^2)$. It is well known (see [2, 3]) that if $N$ is prime, then $\alpha$ is of order $N+1$ or equivalently

$$S_{N+1} \equiv 2 \bmod N.$$

A composite number $N$ satisfying this relation is called a Lucas-pseudoprime for parameter $A$ (Lpsp-$A$). More generally, writing $N+1 = q^t N_0$, one can define the notion of $q$ Lucas pseudoprimes. By analogy with the preceding section, we would like $\alpha^{(N+1)/2} = -1$. For this, we write $\alpha = \gamma^2$ in $F$. The norm of $\gamma$ is $\gamma^{N+1} = \alpha^{(N+1)/2} = -1$. Then the minimal polynomial of $\gamma$ is

$$X^2 - cX - 1$$

with $c$ in $\mathbb{Z}/N\mathbb{Z}$. We write:

$$\gamma^2 - c\gamma - 1 = 0$$

or

$$(\gamma^2 - 1)^2 = c^2\gamma^2$$

which reads

$$(\alpha - 1)^2 = c^2\alpha$$

yielding

$$\alpha^2 + 1 - 2\alpha = c^2\alpha$$

and using the fact that $\alpha^2 + 1 = A\alpha$, one gets

$$A - 2 = c^2$$

in $\mathbb{Z}/N\mathbb{Z}$. This means that $A - 2$ is a quadratic residue modulo $N$.

# 4 Getting free square-roots modulo $p$

Let $p$ be an odd prime. The aim of this section is to show how to find square-roots modulo $p$ as by-products of other calculations.

## 4.1 By-products of $q$-strong tests

Let $a$ be such that $a$ is a square modulo $p$. Then, if $p \equiv 3 \bmod 4$, a square-root of $a$ is given by

$$a^{(p+1)/4} \bmod p.$$

If $p \equiv 5 \bmod 8$, then 2 is a non-residue modulo $p$, therefore $2a$ is not a square. Put

$$\xi = (2a)^{(p-5)/8} \bmod p.$$

Then

$$\xi^2(2a) \equiv (2a)^{(p-1)/4} \equiv i \bmod p$$

where $i^2 \equiv (2a)^{(p-1)/2} \equiv -1$. We also deduce that $\xi a(i-1)$ is a square-root of $a$ since

$$(\xi a(i-1))^2 = \xi^2 a^2 (i-1)^2 \equiv a \bmod p.$$

In this process, we were able to identify $\sqrt{-1} \bmod p$ as well as $\sqrt{a} \bmod p$.
Another way of getting square-roots uses Gaussian periods. For example, take $q = 7$. Let $\zeta$ be a primitive $q$-th root of unity (over $\mathbb{C}$). Define the two periods:

$$\eta_0 = \sum \zeta^{\mathcal{R}}, \ \eta_1 = \sum \zeta^{\mathcal{N}}$$

where $\mathcal{R}$ runs through the quadratic residues modulo $q$, and $\mathcal{N}$ through the non-residues. Then, it is well known (see e.g., [4]) that

$$\eta_0 + \eta_1 = -1, \ \eta_0 - \eta_1 = 2\eta_0 + 1 = \sqrt{(-1)^{(q-1)/2} q}.$$

Coming back to our problem, we replace $\zeta$ by $B$, a root of unity modulo $p$ (i.e., a number $B \neq 1$ such that $B^q \equiv 1 \bmod p$), and get that

$$2(B + B^2 + B^4) + 1 \equiv \sqrt{-7} \bmod p.$$

## 4.2 By-products of Lucas sequences

We use the notations of section 2. In particular, $A-2$ is not a square modulo $p$ and $\alpha^{(p+1)/2}=-1$ in $F=\mathrm{GF}(p^2)\simeq(\mathbb{Z}/p\mathbb{Z})[X]/(X^2-AX+1)$.
Suppose first that $p\equiv 3 \bmod 4$. Then

$$S_{(p+1)/4}\equiv 0 \bmod p$$

since $S_{(p+1)/2}=-2=S^2_{(p+1)/4}-2$. Moreover

$$S_{(p+5)/4}\equiv\sqrt{4-A^2}.$$

We check this with

$$\begin{aligned} S^2_{(p+5)/4} &= (\alpha^{(p+5)/4}+\alpha^{-(p+5)/4})^2=\alpha^{(p+5)/2}+\alpha^{-(p+5)/2}+2\\ &= (-1)\alpha^2+(-1)\alpha^{-2}+2=-(\alpha-1/\alpha)^2=4-A^2 \end{aligned}$$

using the fact that $\alpha^{(p+1)/2}=-1$.
If $p\equiv 7 \bmod 8$, then

$$S_{(p+1)/8}\equiv\sqrt{2} \bmod p$$

using the fact that $S_{(p+1)/4}=0=S^2_{(p+1)/8}-2$.
If $p\equiv 3 \bmod 8$, we may write

$$\begin{aligned} -2S^2_{(p+5)/8} &= -2(\alpha^{(p+5)/4}+\alpha^{-(p+5)/4}+2)=-2(\sqrt{4-A^2}+2)\\ &= -4-2\sqrt{4-A^2}=(\sqrt{A-2}-\sqrt{-A-2})^2 \end{aligned}$$

so that

$$\sqrt{-2}S_{(p+5)/8}=\sqrt{A-2}-\sqrt{-A-2}.$$

When $p\equiv 1 \bmod 4$, we can show that

$$S_{(p-1)/4}\equiv\sqrt{2-A}.$$

This comes from the fact that:

$$S^2_{(p-1)/4}=\alpha^{(p-1)/2}+\alpha^{-(p-1)/2}+2=\alpha^{-1}\alpha^{(p+1)/2}+\alpha\alpha^{-(p+1)/2}+2=-(\alpha+1/\alpha)+2=2-A.$$

We can also use Gaussian periods. Let $q$ be an odd prime and

$$\theta=\alpha^{(p+1)/q}$$

be a primitive $q$-th root of unity in $F$ (i.e., $\theta\neq 1$). Then, using $\eta_0$ and $\eta_1$, one has:

$$\eta_0-\eta_1=\sqrt{(-1)^{(q-1)/2}q}.$$

We must distinguish two cases. The first one corresponds to $q\equiv 1 \bmod 4$. Then $\eta_0$ is in $\mathbb{Z}/p\mathbb{Z}$ and we get $\sqrt{q}$ as usual. For example, taking $q=5$, one has

$$\eta_0=\theta+\theta^4=\theta+\theta^{-1}=S_{(p+1)/q}.$$

On the other hand, when $q\equiv 3 \bmod 4$, $\eta_0$ is in $F$. Put $\omega=\sqrt{\Delta}=\alpha-1/\alpha$. Then $\omega(\eta_0-\eta_1)$ is in $\mathbb{Z}/p\mathbb{Z}$. For instance, if $q=7$, one has

$$\omega(\eta_0-\eta_1)=\omega(\theta-\theta^{-1})+\omega(\theta^2-\theta^{-2})+\omega(\theta^4-\theta^{-4}).$$

We then use the fact that

$$\omega(\theta^i-\theta^{-i})=(\alpha-1/\alpha)(\theta^i-\theta^{-i})=\alpha\theta^i+\alpha^{-1}\theta^{-i}-(\alpha\theta^{-i}+\alpha^{-1}\theta^i)=S_{i(p+1)/q+1}-S_{i(p+1)/q-1}.$$

## 5 Pseudoprimality and square-roots

Suppose we suspect that a given odd integer $N$ is prime. Then, we might try to get square-roots of some numbers. If we can find two square-roots of a number $Z$ that are different, we can factor $N$, since

$$X_1^2 \equiv X_2^2 \bmod N \Rightarrow \gcd(X_1 - X_2, N) \mid N.$$

For instance, if $N \equiv 3 \bmod 4$ is a $\mathrm{spsp}_q(a)$ and a Lpsp-$A$, one can try to find $A$ such that $4 - A^2$ is a quadratic residue modulo $N$. Then, we compute $\sqrt{4 - A^2}$ in two ways, using $A^{(N+1)/4}$ and $S_{(N+5)/4}$ and try to factor $N$ with it.
There is another application of this. The ECPP algorithm [1] requires the computation of square-roots of small numbers modulo $N$, $N$ a probable prime. One can use the same ideas to get these square-roots as free, using the same method and thus speeding the whole process.

## References

[1] A. O. L. Atkin and F. Morain. Elliptic curves and primality proving. Research Report 1256, Institut National de Recherche en Informatique et en Automatique, June 1990. Submitted to *Math. Comp.*

[2] J. Brillhart, D. H. Lehmer, and J. L. Selfridge. New primality criteria and factorizations of $2^m \pm 1$. *Math. Comp.*, 29(130):620–647, 1975.

[3] H. Cohen and A. K. Lenstra. Implementation of a new primality test. *Math. Comp.*, 48(177):103–121, 1987.

[4] H. Davenport. *Multiplicative number theory*, volume 74 of *Graduate Texts in Mathematics*. Springer–Verlag, 2nd edition, 1980.

# 31

# Nombres de Carmichael

Daniel Guillaume
Vélizy, France

[résumé par Philippe Dumas]

*Le nombre de Carmichael est le Canada Dry du nombre premier. Il en a le goût, il en a l'odeur, mais il n'est pas premier.*

D'après le petit théorème de Fermat, si $p$ est un nombre premier et $a$ un entier, le nombre $a^p - a$ est divisible par $p$. Ceci donne une condition nécessaire pour que $p$ soit premier, mais la réciproque est fausse et certains nombres composés passent ce test de primalité. Comme l'a remarqué Lucas, le nombre $2^{341}$ est congru à 2 modulo 341 alors que 341 égale $11 \times 31$ et ceci fournit un exemple de nombre pseudopremier en base 2. Qui plus est, il existe des nombres qui sont pseudopremiers dans n'importe quelle base, comme $561 = 3 \times 11 \times 17$, qui vérifie $a^{561} \equiv a \bmod 561$ pour tout entier $a$. Ces nombres sont les nombres de Carmichael [3], qui les a définis en 1909, et 561 est le plus petit d'entre eux.

## 1 Propriétés des nombres de Carmichael

Pour décrire les nombres de Carmichael, il faut introduire deux outils : l'indicateur d'Euler, $\varphi$, et la fonction de Carmichael, $\lambda$. Pour un entier $N \geq 1$, son indicateur d'Euler, $\varphi(N)$, est le nombre d'entiers compris entre 1 et $N$ et premiers avec $N$. On peut aussi dire que $\varphi(N)$ est l'ordre du groupe des entiers inversibles modulo $N$. Par ailleurs Carmichael a défini [2] la fonction $\lambda$, qui porte son nom, de la façon suivante. Elle coïncide avec $\varphi$ sur les nombres primaires $p^\alpha$, si $p$ est un nombre premier impair ou si $p$ égale 2 et $\alpha$ est inférieur ou égal à 2. Par contre $\lambda(2^\alpha) = 2^{\alpha-2}$ si $\alpha$ vaut au moins 3. La propriété remarquable de $\lambda(N)$ est d'être le plus petit exposant $\Lambda > 0$ tel que $a^\Lambda \equiv 1 \bmod N$ pour tout entier $a$ premier avec $N$. Autrement dit $\lambda(N)$ est le ppcm des ordres des inversibles modulo $N$, ce qui fait que $\lambda(N)$ divise $\varphi(N)$.
Revenons aux nombres de Carmichael : dire que $C$ est un tel nombre signifie que $C$ est composé et que

$$a^{C-1} \equiv 1 \bmod C$$

pour tout entier $a$ premier avec $C$. Ceci s'énonce encore : pour tout diviseur premier $p$ de $C$, le nombre $p - 1$ divise $C - 1$. On en tire les propriétés fondamentales suivantes pour un nombre de Carmichael $C$ :

- $C$ est impair,

- $C$ est sans carré,
- $C$ a au moins trois facteurs premiers,
- si $p$ et $q$ sont deux facteurs premiers de $C$, alors $p$ n'est pas congru à 1 modulo $q$,
- $\lambda(C)$ divise $C-1$.

Cette dernière propriété est caractéristique des nombres de Carmichael. Donnons quelques exemples. Tout d'abord le plus petit nombre de Carmichael, 561, vérifie bien la propriété caractéristique car $\lambda(561) = \mathrm{ppcm}(2,10,16) = 80$ et 80 divise 560. Ensuite [9] les nombres $C = (6m+1)(12m+1)(18m+1)$, où les trois facteurs sont premiers, sont des nombres de Carmichael, car $\lambda(C) = 36m$ divise $C-1 = 36m(36m^2+11m+1)$. Le nombre $1729 = 7\times 13\times 19$ entre dans cette famille. Enfin indiquons la méthode d'extension [4] de Chernik : si $C = k\lambda(C)+1$ est un nombre de Carmichael et si le nombre $p = d\lambda(C)+1$, dans lequel $d$ est un diviseur de $k$, est premier, alors $C' = pC$ est un nombre de Carmichael.

Les deux questions importantes qui se posent sont les suivantes.

- Existe-t-il une infinité de nombres de Carmichael ? Pour essayer de répondre à cette question, on a établi des tables de ces nombres dont la plus importante [8] s'étend jusqu'à $10^{15}$. C. Pomerance, R. Alford et A. Granville [1] viennent de prouver l'infinitude de l'ensemble des nombres de Carmichael. Antérieurement P. Erdős, C. Pomerance et E. Schmutz avaient proposé la conjecture suivante : le nombre $C(x)$ de nombres de Carmichael inférieurs à $x$ vérifie
$$C(x) \underset{x\to\infty}{=} x\exp\left(-\frac{\ln x}{\ln_2 x}\left(\ln_3 x + \ln_4 x + \frac{\ln_4 x - 1}{\ln_3 x} + \cdots\right)\right).$$
Maintenant on sait que
$$C(x) > x^{1/8}$$
pour $x$ assez grand.
- Existe-t-il des nombres de Carmichael avec un nombre de facteurs premiers arbitrairement grand ?

  Jusqu'ici toutes les études s'appuyaient sur la formule $(6m+1)(12m+1)(18m+1)$ et ses variantes, ce qui fournissait au mieux des nombres d'une quinzaine de facteurs. Récemment Zhang [11] a obtenu des nombres de Carmichael à 1300 facteurs et plus de 8300 chiffres décimaux. Le record précédent était tenu par Dubner [5], qui avait exhibé des nombres de 3000 chiffres.

## 2 Recherche de grands nombres de Carmichael

Pour obtenir des nombres $N$ de Carmichael, D. Guillaume et F. Morain [6] partent de la valeur $\Lambda$ de $\lambda(N)$. Ils construisent d'abord un $\Lambda$, hautement composé mais dont l'indicateur d'Euler ne dépasse pas le plus grand entier permis, typiquement $2^{32}$. Puis ils cherchent tous les nombres premiers $p$ tels que $p-1$ divise $\Lambda$, ce qui assure que le ppcm de ces $p-1$ divise $\Lambda$. Ensuite ils calculent tous les produits congrus à 1 modulo $\Lambda$ de ces nombres premiers et obtiennent ainsi des nombres de Carmichael ayant une vingtaine de facteurs.

Donnons un exemple volontairement simplet pour rester lisible. Avec $\Lambda = 2^4 3^2$ on obtient l'ensemble de nombres premiers $\{5, 7, 13, 17, 19, 37, 73\}$ et les deux nombres de Carmichael $1729 = 7 \times 13 \times 19$ et $825265 = 5 \times 7 \times 17 \times 19 \times 73$.
Mais ceci n'est encore pas assez efficace à leurs yeux et ils améliorent leur méthode de différentes façons. Pour éviter de calculer de gros produits, il vaut mieux calculer d'abord le produit complet, $P$, de tous les $p$ puis chercher ensuite des produits partiels à 3, 4 ou 5 facteurs, qui sont congrus à $P$ modulo $\Lambda$. Par simplification, on trouve des nombres de Carmichael à une quarantaine de facteurs. En procédant ainsi, on traite des $\Lambda$ de plus en plus grands et on évite la croissance des données en utilisant le théorème des restes chinois, ce qui permet d'atteindre la centaine de facteurs en quelques heures de temps de calcul.
Pour éviter des essais inutiles, on regarde la forme a priori du dernier facteur utilisé dans le produit partiel et on ne teste que les nombres premiers $p$ qui vérifient cette congruence. Ceci permet d'atteindre le millier de facteurs. Évidemment on peut pousser cette idée plus loin et regarder la forme a priori des deux derniers facteurs $p$ du produit partiel, mais ceci complique la programmation. On peut ainsi obtenir des nombres de Carmichael ayant presque 2000 facteurs en 24 heures sur une station SPARC (pourvue de 50 Mo de mémoire). A l'aide de quelques heuristiques supplémentaires, D. Guillaume et F. Morain [6] ont obtenu leur record de **3075 facteurs et 21163 chiffres décimaux**.

## 3 Conclusion

Revenons sur la démarche employée. Appelons $t$ le nombre d'entiers premiers $p$, pour lesquels $p-1$ divise le nombre $\Lambda$ choisi, et notons $S_u$ l'ensemble des produits partiels de $u$ facteurs $p$ pris dans ces $t$ nombres premiers. Si l'un de ces produits partiels vaut 1 modulo $\Lambda$, on trouve un nombre de Carmichael. On peut aussi demander que ces produits partiels de $u$ facteurs fournissent tous les résidus inversibles modulo $\Lambda$. Pour ceci il est nécessaire que le binomial $\binom{t}{u}$ vérifie

$$\binom{t}{u} > \varphi(\Lambda). \tag{1}$$

En pratique, il semble que cette condition soit suffisante et que les résidus soient également répartis modulo $\Lambda$, ce qui signifie que l'on a une chance sur $\varphi(\Lambda)$ d'obtenir le résidu 1 et donc un nombre de Carmichael à $u$ facteurs. Cette constatation empirique amène D. Guillaume et F. Morain à conjecturer que[7]

$$S_u = (\mathbf{Z}/\Lambda\mathbf{Z})^\times$$

si l'inégalité (1) est vérifiée pour un $u > 2$. Cette conjecture est étroitement liée au problème de Davenport :

> Si $G$ est un groupe abélien, il existe un entier $d$ tel que pour tout $s > d$ et toute suite $g_1, \ldots, g_s$ d'éléments de $G$, l'un des produits partiels $g_{i_1} g_{i_2} \cdots g_{i_l}$ $(i_1 < i_2 < \cdots < i_l$, $1 \leq l \leq s)$ soit égal à 1.

[7] On note $A^\times$ le groupe des éléments inversibles d'un anneau $A$.

Cependant il faut remarquer que les bornes connues sont trop grandes pour être utilisables ici. Précisément van Emde Boas et Kruyswijk [10] ont montré que

$$d \leq m \left(1 + \log \frac{|G|}{m}\right)$$

en notant $m$ le ppcm des ordres des éléments de $G$. Avec $G = (\mathbf{Z}/\Lambda\mathbf{Z})^\times$ on voit que si le nombre $t$ vérifie

$$t > \lambda(\Lambda) \left(1 + \log \frac{\varphi(\Lambda)}{\lambda(\Lambda)}\right) \tag{2}$$

alors on peut affirmer que $\Lambda$ fournit un nombre de Carmichael. Mais l'inégalité (2) n'a pas lieu en général.
L'idée de Pomerance *et alii* [1] est de faire croître $t$ en modifiant peu la borne de (2). Pour cela on remplace un $p$ par de nouveaux facteurs premiers $q$ tels que $q-1$ divise $p\Lambda$. Autrement dit on a changé $\Lambda$ en $p\Lambda$ et le majorant de (2) n'est pas trop modifié car $\lambda(p\Lambda) = \lambda(\Lambda)$. On parvient ainsi à inverser l'inégalité pour obtenir (2) puis l'infinitude de l'ensemble des nombres de Carmichael.

Signalons enfin que la méthode utilisée dans cette construction de grands nombres de Carmichael peut s'étendre à d'autres classes de nombres pseudopremiers comme les nombres $\Delta$-Lucas pseudopremiers, les nombres pseudopremiers elliptiques et permet d'étudier le problème de Williams, le problème de Giuga, par exemple. Ces différents points figurent dans [7].

## Références

[1] W. R. Alford, A. Granville, and C. Pomerance. There are infinitely many Carmichael numbers. In preparation, February 1992.

[2] R. D. Carmichael. Note on a new number theory function. *Bull. AMS*, XVI:232–238, 1910.

[3] R. D. Carmichael. On composite numbers $P$ which satisfy the Fermat congruence $a^{P-1} \equiv 1 \bmod P$. *American Mathematical Monthly*, XIX:22–27, 1912.

[4] J. Chernick. On Fermat's simple theorem. *Bull. AMS*, 45:269–274, April 1939.

[5] H. Dubner. A new method for producing large Carmichael numbers. *Math. Comp.*, 53(187):411–414, July 1989.

[6] D. Guillaume and F. Morain. Building Carmichael numbers with a large number of prime factors. Research Report LIX/RR/92/01, Ecole Polytechnique–LIX, February 1992.

[7] D. Guillaume and F. Morain. Carmichael-like numbers with a large number of prime factors. Preprint, submitted to Eurocrypt'92, January 1992.

[8] R. Pinch. The Carmichael numbers to $10^{15}$. In preparation, January 1992.

[9] P. Ribenboim. *The book of prime number records.* Springer, 2nd edition, 1989.

[10] P. van Emde Boas and D. Kruyswijk. A combinatorial problem on finite abelian groups, III. Technical Report ZW–008, Math. Centrum Amsterdam Afd. Zuivere Wisk., 1969.

[11] M. Zhang. Searching for large Carmichael numbers. Submitted to Mathematic of Computation, December 1991.

# 32

# Algorithmes pour la conception de circuits arithmétiques rapides

Jean-Michel Muller
ENS, Lyon

[résumé par Valérie Ménissier-Morain & François Morain]

## 1 Introduction

Deux thèmes sont abordés: l'arithmétique "en ligne" (c'est un mode de calcul dans lequel les opérandes circulent en série, poids forts en tête, et sont représentés dans un système redondant d'écriture des nombres), et les algorithmes "de type Cordic" de calcul des fonctions élémentaires.

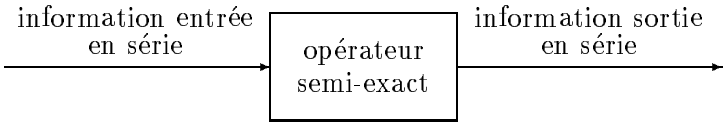
## 2 L'arithmétique "en ligne"

L'arithmétique "en ligne" est un moyen de calcul qui permet d'atteindre de hautes performances en permettant un "pipe-line" au niveau du chiffre.
Le sens de circulation "poids forts en tête" des chiffres est plus naturel que le sens "poids faibles en tête", car il correspond à la notion de continuité (on va du grossier au précis) et fournit un contrôle d'erreur naturel. De surcroît, le sens "poids faibles en tête" ne permettrait pas d'effectuer des divisions ou de calculer des fonctions élémentaires.
Deux questions se posent: quel système d'écriture des nombres utiliser et quels algorithmes, adaptés à quelles architectures, choisir pour les opérations arithmétiques ($+$, $-$, $\times$, $/$) et les fonctions transcendantes?
Il existe divers types d'arithmétique:

**exacte:** entiers, rationnels, manipulation symbolique d'objets non finis,

**approchée:** c'est l'arithmétique flottante usuelle,

**semi-exacte:** cette arithmétique fonctionne suivant le schéma suivant:




L'idée du calcul en série est de ne consommer qu'un grain d'information de chaque opérande pour produire un grain d'information en sortie.

Il y a deux avantages à cela: d'une part le *layout* se trouve simplifié et d'autre part cela permet un "pipe-line" au niveau du chiffre et, par conséquent, un enchaînement rapide des calculs.
Mais ce système ne marche que si les chiffres circulent *toujours* dans le même sens.
Hélas, les algorithmes classiques d'addition et de multiplication produisent les chiffres de droite à gauche à cause de la propagation des retenues, alors que celui de la division produit les chiffres de gauche à droite.

## 2.1 Représentation d'Avizienis et algorithmes d'addition

On utilise alors une représentation redondante des nombres due à Avizienis ([1] et [4] pp. 34-40): on travaille dans une base $B$ et les chiffres sont des entiers entre $-a$ et $+a$, où $a$ est un chiffre en base $B$ ($1 \leq a \leq B-1$).
Avizienis a établi deux résultats:

- *on peut écrire tous les nombres entiers dans cette représentation si et seulement si* $2a+1 \geq B$,
- *si* $2a \geq B+1$, *alors il existe un algorithme pour effectuer des additions de manière* complètement parallèle, *sans propagation de retenue.*

Voici une description de cet algorithme d'addition:

$$\begin{aligned} t_{i+1} &= \begin{cases} -1 & \text{si} \quad x_i + y_i \leq -a \\ +1 & \text{si} \quad x_i + y_i \geq a \\ 0 & \text{si} \quad -a+1 \leq x_i + y_i \leq a-1 \end{cases} \\ w_i &= x_i + y_i - B t_{i+1} \\ s_i &= w_i + t_i \end{aligned}$$

avec $w_n = t_0 = 0$.
On montre que cet algorithme ne produit jamais de retenue.
Les deux conditions $a \leq B-1$ et $2a \geq B+1$ ne peuvent être satisfaites que si $B$ est supérieur ou égal à 3. La méthode d'Avizienis ne peut donc être utilisée que dans ce cas-ci.
Il est possible de faire également des additions de manière totalement parallèle (c'est-à-dire en un temps totalement indépendant de la taille des opérandes) en base 2, mais la méthode utilisée est alors plus complexe que celle d'Avizienis [2]. Cet algorithme a pour délai 2 (c'est-à-dire le nombre de grains d'information sur l'opérande à fournir avant de produire un grain d'information pour le résultat), qui est le délai minimum pour l'addition en base 2.

## 2.2 Algorithmes de multiplication et de division "en ligne"

Ercegovac et Trivedi ([3] et [4] pp. 134-135 pour la multiplication, pp. 161-164 pour la division) ont proposé en 1977 des algorithmes de multiplication et division "en ligne" de délai 5.

## 2.3 Que peut-on calculer en ligne?

Remarquons tout d'abord que la calculabilité nécessite la continuité de l'opérateur, ceci vient du fait que l'on calcule avec des systèmes redondants d'écriture des nombres.

- On sait calculer avec un opérateur indépendant de la taille des nombres manipulés (*un automate fini*): $+$, $-$, max, min, et toute application affine avec des constantes rationnelles,
- On sait calculer avec un opérateur "linéaire" constitué d'un même élément répliqué un nombre de fois égal à la taille des nombres manipulés: $\times$, $/$, $\sqrt{}$, et tout polynôme à coefficients rationnels ou son inverse,
- On sait calculer avec des opérateurs constitués d'une partie "linéaire" et d'une table contenant un nombre de constantes proportionnel à la taille des nombres manipulés: sin, cos, arctan, exp et log.

## 3 Les algorithmes "de type Cordic"

Les algorithmes "de type Cordic" [5], bien qu'assez anciens (le premier algorithme de cette classe a trois siècles et a été inventé par Briggs, un contemporain de Neper, pour construire la première table de logarithmes), sont un des moyens de calcul des fonctions élémentaires (sinus, exponentielle, logarithme, ...) les plus employés (on trouve des algorithmes de ce type dans la plupart des calculatrices, et dans des coprocesseurs comme l'Intel 8087 ou le Motorola 68881).
A titre d'exemple, considérons le problème du calcul des fonctions circulaires élémentaires cosinus et sinus. Soit $\theta$ un réel, $0 \leq \theta \leq \pi/2$. Pour calculer $(\cos\theta, \sin\theta)$, il suffit de faire une rotation d'angle $\theta$ à partir du point $(1,0)$ et de lire les coordonnées du point d'arrivée. L'idée est alors de décomposer cette rotation en petites rotations d'angle $d_n e_n$ avec $d_i = \pm 1$, de sorte que les rotations d'angle $\pm e_i$ soient faciles à effectuer et que bien sûr $\theta = \sum_{i=0}^{\infty} d_i e_i$. On pose $\theta_n = \sum_{i=0}^{n} d_i e_i$. A la $n$-ième étape, on tournera d'un angle $+e_n$ si $\theta_n \leq \theta$, et de $-e_n$ sinon. On montre alors le résultat suivant :

*Si $(e_n)$ est une suite décroissante et sommable de réels positifs, vérifiant*

$$\forall n, \quad e_n \leq \sum_{i=n+1}^{\infty} e_i$$

*alors, pour tout $\theta \in [-\sum_{i=0}^{\infty} e_i, +\sum_{i=0}^{\infty} e_i]$, la suite $(\theta_n)$ définie par*

$$\begin{array}{lcl} \theta_0 & = & 0 \\ d_n & = & 1 \text{ si } \theta_n \leq \theta, -1 \text{ sinon} \\ \theta_{n+1} & = & \theta_n + d_n e_n \end{array}$$

*a pour limite $\theta$ quand $n$ tend vers l'infini.*

L'algorithme de calcul de $(\cos\theta, \sin\theta)$ se fait alors en prenant pour suite $(e_n)$ la suite de terme général $e_n = \arctan(2^{-n})$. Écrivons :

$$V_n = \begin{pmatrix} x_n \\ y_n \end{pmatrix} = \begin{pmatrix} \cos\theta_n \\ \sin\theta_n \end{pmatrix}.$$

On passe de $V_n$ à $V_{n+1}$ par une rotation d'angle $d_n e_n$. Autrement dit :

$$\begin{pmatrix} x_{n+1} \\ y_{n+1} \end{pmatrix} = \begin{pmatrix} \cos(d_n e_n) & -\sin(d_n e_n) \\ \sin(d_n e_n) & \cos(d_n e_n) \end{pmatrix} \begin{pmatrix} x_n \\ y_n \end{pmatrix}$$

ce qui conduit, avec le choix particulier de $e_n = \arctan(2^{-n})$ à

$$V_{n+1} = \frac{1}{\sqrt{1+2^{-2n}}} \begin{pmatrix} 1 & -d_n 2^{-n} \\ d_n 2^{-n} & 1 \end{pmatrix} V_n.$$

Quand $n$ tend vers l'infini, on obtient

$$V_\infty = \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix} = \prod_{n=0}^{\infty} \frac{\begin{pmatrix} 1 & -d_n 2^{-n} \\ d_n 2^{-n} & 1 \end{pmatrix}}{\sqrt{1+2^{-2n}}} \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} = K \prod_{n=0}^{\infty} \begin{pmatrix} 1 & -d_n 2^{-n} \\ d_n 2^{-n} & 1 \end{pmatrix} V_0.$$

L'algorithme de Cordic consiste à calculer la suite $V'_n$ définie par

$$V'_0 = \begin{pmatrix} 1/K \\ 0 \end{pmatrix},$$

$$V'_{n+1} = \begin{pmatrix} x'_{n+1} \\ y'_{n+1} \end{pmatrix} = \begin{pmatrix} 1 & -d_n 2^{-n} \\ d_n 2^{-n} & 1 \end{pmatrix} V'_n$$

qui converge par construction vers

$$V'_\infty = V_\infty = \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix}.$$

On montre que de petites modifications de ces algorithmes permettent l'utilisation de systèmes redondants d'écriture des nombres, et autorisent ainsi d'excellentes performances de calcul. On peut notamment utiliser ces algorithmes modifiés pour calculer "en ligne" (méthodes de Takagi ; améliorations de l'auteur).

## Références

[1] A. Avizienis. Signed-digit number representations for fast parallel arithmetic. *IRE Transactions on electronic computers*, 10:pp. 389–400, 1961.

[2] C. Y. Chow and J. E. Robertson. Logical design of a redundant binary adder. In *Proceedings of the* $4^{th}$ *symposium on computer arithmetic*, pages pp. 109–115, October 1978.

[3] M. D. Ercegovac and K. S. Trivedi. On-line algoritms for division and multiplication. *IEEE Transactions on Computers*, C-26:pp. 681–687, July 1977.

[4] J. M. Muller. *Arithmétique des ordinateurs*. études et recherches en informatique. Masson, 1989.

[5] J. Volder. The CORDIC computing technique. *IRE Transactions on Computers*, September 1959.

# 33

# Circuits synchrones, nombres 2-adiques, et codages RSA

Jean Vuillemin
Digital PRL, Rueil Malmaison

[résumé par Paul Zimmermann]

L'exposé traite de l'arithmétique 2-adique et de ses propriétés, notamment vis-à-vis des circuits combinatoires. On montre l'équivalence entre un nombre 2-adique et un fil d'un circuit, entre une fonction sur les nombres 2-adiques et un circuit. On montre aussi que les différents types de circuits (combinatoire, synchrone) correspondent à des classes de fonctions (continues, strictes, en ligne). L'arithmétique $p$-adique a été utilisée pour un codage rapide d'informations par l'algorithme RSA.

## 1 Nombres $p$-adiques

Un rationnel 2-adique a une notation de la forme suivante

$$R = r_{-v} \dots r_0.r_1 \dots r_n \dots$$

qui représente le nombre $\sum_{k \geq -v} r_k 2^k$. Par exemple, $0.101(110)^*$ représente $2 + 8 + 16(3 + 3 \cdot 8 + 3 \cdot 8^2 + \cdots) \simeq 10 + 16 \cdot 3 \cdot 1/(1 - 8) = 10 - 48/7 = 22/7$. A l'opposé de l'arithmétique réelle, qui représente les nombres des poids forts vers les poids faibles, l'arithmétique 2-adique va des poids faibles vers les poids forts, et est donc en quelque sorte duale de l'arithmétique réelle.
Si $p$ est un entier premier, tout nombre rationnel a une représentation $p$-adique : $\mathbb{Q} \subset \mathbb{Q}_p$. La représentation $p$-adique des rationnels est ultimement périodique, et peut donc s'écrire de manière finie sous la forme $r_{-v} \dots r_0.r_1 \dots r_k(r_{k+1} \dots r_{k+l})^*$ où $r_{-v} \dots r_0.r_1 \dots r_k$ représente la partie positive $r_{-v}p^{-v} + \cdots + r_0 + r_1 p + \cdots + r_k p^k$ et $(r_{k+1} \dots r_{k+l})^*$ la partie négative $(r_{k+1}p^{k+1} + \cdots + r_{k+l}p^{k+l})/(1 - p^l)$.
Ainsi dans $\mathbb{Q}_2$, $-16$ s'écrit $0.000(1)^*$, $-6/7$ s'écrit $0.11(011)^*$, $-2/3$ s'écrit $0.1(01)^*$. La procédure MAPLE ci-dessous calcule une forme 2-adique d'un rationnel $q$ :

```
twoadic := proc(q) # returns a list [v,[r_{-v}, r_{-v+1}, ...]]
local a,b;
    if q<0 then # q = a - 2^b with 2^b>=|q|
            b:=ceil(log(-q)/log(2));
            a:=q+2^b;
            add(procname(a),[-b,[[1]]])
    elif type(q,nonnegint) then [0,convert(q,base,2)]
    elif denom(q) mod 2 = 0 then div2(procname(2*q))
    else # q = a/b with a and b odd : q = a/(1+2*c)
        odddiv(numer(q),(denom(q)-1)/2)
    fi;
end:
```

où `add` calcule la somme de deux représentations 2-adiques, et `odddiv(a,b)` calcule le quotient 2-adique de $a$ par $1+2b$.

```
> twoadic(1),twoadic(-1),twoadic(-6/7);

            [0, [1]],   [0, [[1]]],   [0, [0, 1, 1, 0, [1, 1, 0]]]
```

De la même façon, il est possible de retrouver le rationnel correspondant à un nombre 2-adique :

```
value := proc(x)
local v,l,s,t;
    v:=op(1,x); l:=op(2,x); s:=0; t:=2^(-v);
    while l<>[] and not type(l[1],list) do
        s := s + l[1]*t;
        l := subsop(1=NULL,l);
        t := 2*t;
    od;
    if l=[] then s
    else
        l := l[1];
        s + sum(l[i]*2^(i-1),i=1..nops(l))*t/(1-2^nops(l))
    fi
end:

> value(twoadic(355/113)), value(twoadic(-3/97));

                                    355
                                    ---, -3/97
                                    113
```

**Proposition 1** *Les rationnels 2-adiques forment un corps* $\mathbb{Q}_2 = \{0, 1, +, -, \times, /\}$ *admettant* $\mathbb{Q}$ *comme sous-corps. Les entiers 2-adiques forment un anneau* $\mathbb{Z}_2 = \{0, 1, +, -, \times\}$ *admettant* $\mathbb{Z}$ *et* $\mathbb{Z}/(1+2\mathbb{Z})$ *comme sous-anneaux.*

## 2 Circuits

Il est montré dans cette partie un lien direct entre les nombres 2-adiques et les circuits logiques synchrones. Plus précisément, on verra qu'on peut associer à tout fil d'un circuit un nombre 2-adique, que pour tout rationnel 2-adique, on peut construire un circuit le "calculant", et qu'il existe un lien entre les fonctions calculables par des circuits synchrones et les fonctions continues.

**Définition 1** *(Circuit digital) La valeur de toute variable d'un circuit digital C est un bit qui ne peut changer qu'à des temps entiers :*

$$\forall v \in \mathcal{V}(C), t \in \mathbb{R}, v^t = v^{\lfloor r \rfloor} \in \{0, 1\}$$

*où* $v^t$ *représente la valeur du point v du circuit à l'instant t.*

Ainsi, chaque point d'un circuit (ou fil, puisque la tension est la même en tout point d'un fil) est mis en correspondance avec un nombre 2-adique $v^0.v^1 \ldots v^n \ldots$ De la même façon, les composants d'un circuit sont mis en correspondance avec des opérations sur les nombres 2-adiques :

**Définition 2** *(Multiplexeur et circuit combinatoire) Le multiplexeur noté ? est une fonction de $\mathbb{Z}_2^3$ dans $\mathbb{Z}_2$, définie par*

$$?(c\ t\ f) = m \qquad \textit{tel que} \qquad m = \begin{cases} t & \textit{si } c = 1 \\ f & \textit{si } c = 0. \end{cases}$$

*Un circuit combinatoire est un circuit comportant des entrées $i_0, \ldots, i_n, \ldots$, des sorties $o_0, \ldots, o_n, \ldots$, des multiplexeurs, et tel qu'il existe un ordre total sur les fils des multiplexeurs :*

$$\forall ?(c\ t\ f) \Rightarrow m, \quad c, t, f < m.$$

Par exemple, le circuit suivant est un circuit combinatoire effectuant l'addition de trois bits $a, b, c$ avec retenue. Il comprend cinq multiplexeurs, $(a+b+c) \bmod 2$ est mis dans $s$ et $(a+b+c)$ div 2 dans $r$ :

$$?(b\ 0\ 1) \Rightarrow \bar{b}, \quad ?(a\ \bar{b}\ b) \Rightarrow x, \quad ?(c\ 0\ 1) \Rightarrow \bar{c}, \quad ?(x\ c\ b) \Rightarrow r, \quad ?(x\ \bar{c}\ c) \Rightarrow s.$$

Il est intéressant de constater le lien entre les fonctions calculables par des circuits combinatoires et les fonctions continues :

**Théorème 1** *Les assertions suivantes sont équivalentes :*

1. *$f(i_0, \ldots, i_n, \ldots) \Rightarrow (o_0, \ldots, o_n, \ldots)$ est calculable par un circuit combinatoire.*
2. *La fonction $f(\sum i_n 2^n) = \sum o_n 2^n$ est continue sur $\mathbb{Z}_2$ avec la norme $|r_v r_{v+1} \ldots|_2 = 2^{-v}$.*
3. *Chaque sortie dépend d'un nombre fini d'entrées.*

Par exemple, le mélange de deux entrées $\pi$ et les projections $\pi_0$ et $\pi_1$ sont des circuits combinatoires, donc représentent des fonctions continues :

$$\begin{aligned} a \bmod 2 + 2\pi(b, a \text{ div } 2) &\Rightarrow \pi(a, b) \\ a \bmod 2 + 2\pi_0(a \text{ div } 4) &\Rightarrow \pi_0(a, b) \\ \pi_0(a \text{ div } 2) &\Rightarrow \pi_1(a, b) \end{aligned}$$

Au passage, on aura remarqué que ces trois fonctions mettent en évidence un isomorphisme entre $\mathbb{N} \times \mathbb{N}$ et $\mathbb{N}$.

## 2.1 Circuits synchrones

**Définition 3** *(Registre et circuit synchrone) Le registre à décalage, noté $2\times$, est une fonction de $\mathbb{Z}_2$ dans $\mathbb{Z}_2$, définie par*

$$2 \times i \Rightarrow r \qquad \textit{tel que} \qquad r^0 = 0, r^{t+1} = i^t.$$

*Un circuit synchrone est un circuit combinatoire comportant en plus des registres, et ayant un nombre fini d'entrées et de sorties.*

L'ajout des registres permet, même avec des entrées constantes, de faire varier les sorties au cours du temps. Ainsi, à chaque fil $v$ est associé le nombre 2-adique $B = v^0.v^1 \ldots v^n \ldots$ Le registre transforme $B$ en $2B$, l'inverseur transforme $B$ en $-1-B$. Ainsi, on peut construire un circuit "calculant" -16, -6/7, -2/3 ou 22/7!

**Théorème 2** *Les assertions suivantes sont équivalentes :*

1. $f(i) \Rightarrow o$ *est calculable par un circuit synchrone.*
2. $f(\sum i^t 2^t) = \sum o^t 2^t$ *est à la fois en ligne* $(\forall B, n, f(B) = f(B \bmod 2^n) \bmod 2^n)$ *et stricte* $(\forall B, f(2B) = 2f(B))$.

Le fait d'être en ligne signifie que les $n$ premiers bits de la sortie ne dépendent que des $n$ premiers bits de l'entrée. Le fait d'être stricte signifie que l'on peut faire du *retiming*, c'est-à-dire que l'on peut faire commuter des registres avec le circuit.
Les circuits courants comme $\cap, \cup, +, -, \pi$ sont stricts. Le ou exclusif, la multiplication, les fonctions $1/(1+2b)$ et $\sqrt{1+8b}$ sont en ligne, mais pas la division par 2 (le $n$-ème bit de la sortie dépend du $(n+1)$-ème de l'entrée), ni $\pi_0$ et $\pi_1$ (le $n$-ème bit de la sortie dépend des bits $2n-1$ et $2n$ de l'entrée).
On peut ainsi fabriquer un circuit synchrone "universel" qui comporte une entrée $B$, des registres à décalage qui calculent $2B$, $4B$, $8B$, ..., des multiplexeurs $F_0$, $F_1$, $F_2$, ..., commandés par une ligne de registres à décalage tels qu'à l'instant $n$, la sortie de $F_n$ soit la sortie du circuit. Il suffit donc de mettre en entrée de $F_n$ un circuit combinatoire idoine ayant $B$, $2B$, $4B$, $\ldots 2^n B$ comme entrées.
Alors que les circuits d'addition et de soustraction sont de taille finie, les circuits de multiplication et de racine carrée sont de taille *infinie*.

Enfin, un circuit en ligne de produit modulaire $A \times B \bmod C$ a été réalisé sur une carte PAM, pour des entiers 4-adiques de 256 bits. Ce circuit est à la base d'un système de codage RSA à 200KB par seconde.

## Références

[1] Y. Amice. *Les nombres p-adiques.* Presses Universitaires de France, 1975.

[2] R. E. Bryant. Graph-based algorithms for boolean function manipulation. *IEEE Transactions on Computers*, 35(8):677–691, 1986.

[3] K. Hensel. *Zahlentheorie.* Göshen, Berlin-Leiptzig, 1913.

[4] N. Koblitz. *p-adic Numbers, p-adic Analysis and Zeta Functions.* Springer-Verlag, 1977.

[5] C. Leiserson and J. Saxe. Retiming synchronous circuitry. *Algorithmica*, 6(1):5–35, 1991.

[6] C. Mead. *Analog VLSI and Neural Systems.* Addison-Wesley, 1989.

[7] J. Vuillemin. On Circuits and Numbers. Preprint.

# 34

# Cryptanalyse différentielle du DES en 16 rounds

Adi Shamir
The Weizmann Institut of Science, Rehovot

[résumé par Abalo Baya]

**DES** (Data Encryption Standard) est sans doute le cryptosystème le plus connu et le plus utilisé dans un grand nombre de transactions civiles. Développé par IBM et adopté par NBS (National Bureau of Standards) dans les années 70, ce cryptosystème a résisté à toutes les attaques publiées jusqu'à présent. L'auteur développe ici une cryptanalyse différentielle des cryptosystèmes à $n$ rounds en général et du DES en particulier.

## 1 Cryptosystèmes à $n$ rounds

Un cryptosystème à $n$ rounds est une fonction cryptographiquement forte basée sur $n$ itérations d'une fonction cryptographiquement faible. Chaque itération est appelée *round* . La fonction itérée (fonction round) est fonction du résultat du round précédent et d'une sous-clé de la clé de codage. La fonction round est habituellement basée sur des S-boîtes, des permutations de bits, des opérations arithmétiques et de l'opération "ou exclusif" (notée par $\oplus$ ou XOR). Les S-boîtes sont des tables de transformation non-linéaire d'un petit segment de bits en un autre petit segment de bits. Les permutations de bits permettent de réarranger les sorties des S-boîtes de telle sorte que les entrées des S-boîtes d'un round proviennent des sorties d'un maximum de S-boîtes du round précédent. L'opération XOR sert à faire le brassage de la sous-clé et du message. Dans la plupart des applications, l'algorithme de codage est supposé connu alors que la clé de codage reste secrète.

Un exemple type de cryptosystème à $n$ rounds ($n = 16$) est le DES. Chacune des clés du DES a 56 bits tandis que les messages en ont 64. La principale partie de la fonction round est une fonction $F$ dont l'argument est un couple ($m_1$, $K_s$), où $m_1$ est une séquence de 32 bits et où $K_s$ est une clé de 48 bits calculés à partir de la clé de codage. L'entrée $m_1$ de 32 bits est transformée (par un extenseur) en une séquence $m_2$ de 48 bits. On fait ensuite le XOR de $K_s$ et de $m_2$. Le résultat de l'opération XOR est transformé par 8 S-boîtes en une séquence $m_3$ de 32 bits et la sortie de $F$ est obtenue par permutation des bits de $m_3$. La spécification complète du DES est exposée dans [7].

## 2 Cryptanalyse différentielle du DES

Une littérature extensive sur le DES a été publiée depuis son adoption par le NBS en 1977. Mais aucune cryptanalyse faisant mieux qu'une recherche exhaustive sur $2^{55}$ clés (on dit que la complexité de l'attaque est de $2^{55}$) n'a pu être réalisée. Ce manque de progrès sur l'amélioration de la complexité concernant la cryptanalyse du DES en 16 rounds a conduit beaucoup de chercheurs à analyser des versions simplifiées du DES et en particulier des versions du DES en moins de 16 rounds. Ainsi Chaum et Evertse [5] ont développé une attaque de complexité $2^{54}$ sur le DES en 6

rounds et ils ont démontré qu'une telle attaque ne pouvait pas être appliquée à un DES de plus de 7 rounds. Par ailleurs l'attaque de Davies [6], de complexité $2^{40}$ sur un DES en 8 rounds, est moins efficace sur un DES en 16 rounds qu'une simple recherche exhaustive. La première version [1] de la cryptanalyse différentielle est efficace sur un DES ne dépassant pas 15 rounds et la version actuelle [4] "casse" le DES en 16 rounds avec une complexité strictement inférieure à $2^{55}$.

La cryptanalyse différentielle est une attaque avec choix de message en clair, l'outil essentiel de cette attaque étant le concept de "paires de messages encryptés" : une paire de messages encryptés est une paire résultant du codage de deux messages ayant une différence spécifique. Fondamentalement, la cryptanalyse différentielle est une méthode qui analyse *l'effet des différences* des paires de messages *sur les différences* des messages encryptés résultant. De telles différences sont utilisées pour attribuer des probabilités aux valeurs possibles des clés, ce qui permet de retenir uniquement les clés les plus probables. Pour la cryptanalyse du DES, la différence spécifique choisie est le XOR de deux messages en clair. En fait on génère un nombre suffisant de paires de messages en clair donnant lieu à des XOR intermédiaires spécifiés par une caractéristique convenablement choisie (voir [1] pour la définition de caractéristique). L'analyse partielle ou totale de ces messages en clair permet d'obtenir les clés les plus probables. Une recherche exhaustive sur ces clés conduit *presque toujours* à la clé secrète désirée. La table suivante résume les résultats obtenus. La dernière colonne de cette table reporte les complexités atteintes dans la première version [1] de la cryptanalyse différentielle.

| Nb. de rounds | nb. de messages choisis | nb. de messages analysés | complexité de de l'analyse | complexité trouvée dans [1] |
|---|---|---|---|---|
| 8 | $2^{14}$ | 4 | $2^{9}$ | $2^{16}$ |
| 9 | $2^{24}$ | 2 | $2^{32}$ | $2^{26}$ |
| 10 | $2^{24}$ | $2^{14}$ | $2^{14}$ | $2^{15}$ |
| 11 | $2^{31}$ | 2 | $2^{32}$ | $2^{36}$ |
| 12 | $2^{31}$ | $2^{21}$ | $2^{21}$ | $2^{43}$ |
| 13 | $2^{39}$ | 2 | $2^{32}$ | $2^{44}$ |
| 14 | $2^{39}$ | $2^{29}$ | $2^{51}$ | $2^{51}$ |
| 15 | $2^{47}$ | $2^{7}$ | $2^{37}$ | $2^{52}$ |
| 16 | $2^{47}$ | $2^{36}$ | $2^{37}$ | $2^{58}$ |

Table 1 : Résultats de la cryptanalyse différentielle du DES.

## Références

[1] E. Biham and A. Shamir. Differential cryptanalysis of DES-like cryptosystems. *Journal of Cryptology*, 4(1):3–72, 1991. The extended abstract appears in Advances in Cryptology, proceedings of CRYPTO 90.

[2] E. Biham and A. Shamir. Differential cryptanalysis of Feal and $N$-Hash. Technical report CS91-17, The Weizmann Institute of Science, 1991. The extended abstract appears in Advances in Cryptology, proceedings of CRYPTO 91.

[3] E. Biham and A. Shamir. Differential cryptanalysis of Snefru, Khafre, REDOC-II, LOKI and Lucifer. Technical report CS91-18, The Weizmann Institute of Science, 1991. The extended abstract appears in Advances of Cryptology, proceedings of CRYPTO 91.

[4] E. Biham and A. Shamir. Differential cryptanalysis of the full 16-round DES. Preprint, December 1991.

[5] D. Chaum and J.-H. Evertse. Cryptanalysis of DES with a reduced number of rounds, Sequences of linear factors in block ciphers. In *Advances in Cryptology*, pages 192–211, 1985. Proceedings of CRYPTO 85.

[6] D. W. Davies. Private communication.

[7] National Bureau of Standards. Data encryption standard. *FIPS*, 46, January 1977. U. S. Department of Commerce.

# 35

# Modelling Primitive Recursive Functions with Exponential Diophantine Equations

Yuri Matijasevich
Steklov Institute, Saint-Petersburg

It is known since 1970 that every partial recursive function $f$ has a representation of the form

$$y = f(x_1, \ldots, x_m) \qquad \text{iff} \qquad \exists z_1 \cdots z_n \quad y = P(x_1, \ldots, x_m, z_1, \ldots, z_n)$$

where $P$ is a polynomial. So far two essentially different techniques were known for constructing such a $P$ for given $f$. One technique was based on arithmetization of $f$ and further elimination of universal quantifiers; the other one proceeded by constructing Turing machine or register machine calculating $f$, and then simulating the machine by Diophantine equations.
The talk is about a direct technique recently discovered by the speaker for constructing such a $P$. Actually, a detailed proof is given for a weaker result (known since 1961 and due to M. Davis, H. Putnam and J. Robinson) with $f$ being primitive recursive and $P$ being an exponential polynomial.

# 36

# Some Investigations on the Riemann Hypothesis with Computers

Yuri Matijasevich
Steklov Institute, Saint-Petersburg

The talk surveys some connections between RH and calculations on computers, abstract and electronic, in particular:
Exact verification of RH for small zeroes on computers with limited precision;
A connection between RH and the P = NP problem;
Position of RH in the arithmetical hierarchy and its heuristic significance;
Proof of Turan's inequalities (a corollary to RH) by calculations.

# Contents