Exemples d'analyses d'algorithmes en Arithmétique, Théorie de l'Information et Fouille de Données

Loïck Lhote

Normandie Université

Habilitation à Diriger des Recherches

Pour obtenir le diplôme d'habilitation à diriger des recherches

Spécialité INFORMATIQUE

Préparée au sein de l'Université de Caen, Normandie

Exemples d'analyses d'algorithmes en Arithmétique, Théorie de l'Information et Fouille de Données.

Présentée et soutenue par
Loïck LHÔTE

HDR soutenue publiquement le 9 décembre 2016 devant le jury composé de		
Frédérique BASSINO	Professeure, LIPN, Université de Paris 13	Rapporteure
Bruno SALVY	DR INRIA, LIP, ENS Lyon	Rapporteur
Hsien-Kuei HWANG	Chercheur, Institute of Statistical Science, Academia Sinica, Taiwan	Rapporteur
Lhouari NOURINE	Professeur, LIMOS , Université Blaise Pascal	Examinateur
Bruno CRÉMILLEUX	Professeur, GREYC, Université de Caen Normandie	Examinateur
Julien CLÉMENT	CR CNRS, HDR, GREYC, Université de Caen Normandie	Examinateur (Directeur de l'HDR)

UNICAEN
UNIVERSITÉ
CAEN
NORMANDIE

GREYC

Sommaire

Introduction

Pendant mon DEA à l'université de Caen (2001-2002), j'ai découvert simultanément le monde de l'informatique et le monde de la recherche. Avec une formation initiale en mathématiques, je me suis naturellement orienté vers l'informatique mathématique. J'ai réalisé mon stage de DEA sous la direction de Brigitte Vallée sur un sujet en lien avec l'analyse dynamique et j'ai ensuite continué en thèse. Très vite, je me suis intéressé à l'analyse des algorithmes du PGCD et le hasard des rencontres a voulu que je partage mon bureau avec des doctorants en fouille de données. De nombreuses discussions plus tard, nous étions convaincus de l'intérêt que peuvent avoir les analyses d'algorithmes en fouille de données. Cette conviction, je la partage toujours et une partie des travaux de ce mémoire s'inscrit dans cette démarche. J'ai ensuite obtenu un poste de maître de conférences à Caen en 2007.

Mes travaux actuels sont en lien avec trois domaines de l'informatique : les algorithmes du PGCD, la fouille de données et la théorie de l'information. Ils partagent tous un fil conducteur commun : l'aléa. L'aléa intervient sous différentes formes dans ce mémoire. Tout d'abord, l'objet étudié peut lui-même être aléatoire comme les sources en théorie de l'information dont nous étudions une grandeur caractéristique appelée taux d'entropie. L'objet étudié peut ne pas être aléatoire mais nous décrivons ses propriétés avec un point de vue probabiliste en le générant aléatoirement. C'est la cas par exemple avec le PGCD de plusieurs entiers ou polynômes, les hypergraphes et les motifs en fouille de données. Si l'objet étudié est un algorithme (algorithmes du PGCD, de réduction de réseaux euclidiens ou de calcul de traverses minimales), nous nous intéressons au comportement probabiliste de ce dernier lorsque les entrées sont générées aléatoirement.

Malgré la diversité des contextes, des objets et des problèmes, j'ai toujours adopté la même approche : celle de l'analyse d'algorithmes et des structures de données. L'analyse d'algorithmes a été introduite par Knuth dans les années 60 et a connu de nombreux développements, notamment en France sous l'impulsion de Philippe Flajolet avec la combinatoire analytique [60]. L'objectif est de décrire au mieux le comportement des algorithmes et des structures lorsque la taille des entrées augmente. Les analyses présentées dans ce mémoire reposent souvent sur la combinatoire analytique ou l'analyse dynamique qui s'appuient sur des manipulations formelles et analytiques de séries génératrices. Un chapitre est consacré à ces méthodes.

Au cours de ma *jeune* carrière, j'ai aussi été amené à co-encadrer des doctorants. Avec Brigitte Vallée et Julien Clément, j'ai co-encadré la thèse de Mariya Georgieva pendant ses deux dernières années de thèse (2011-2013). Ce fut une expérience enrichissante et d'un point de vue scientifique, cela m'a ouvert aux problématiques de la réduction des réseaux euclidiens. Depuis 2015, je co-encadre avec Julien Clément la thèse de Dimitri Darthenay. Au début, le sujet était orienté vers une généralisation des travaux de Mariya et actuellement, Dimitri s'intéresse à l'analyse des algorithmes de tri et de recherche.

Plan du mémoire. Le chapitre 1 présente succinctement les principes et les méthodes d'analyses d'algorithmes. La combinatoire analytique et l'analyse dynamique y sont aussi dé-

crites. Ces méthodes reposent sur la manipulation de séries génératrices dont la nature (séries ordinaires ou de Dirichlet) dépend du contexte. L'analyse dynamique s'appuie sur la combinatoire analytique mais elle combine les séries génératrices avec des opérateurs issus de la théorie des systèmes dynamiques. Le chapitre 1 décrit les liens entre tous ces objets.

Les chapitres 2 et 3 sont reliés directement avec les algorithmes du PGCD. Depuis plusieurs années, l'analyse dynamique a permis de bien comprendre les algorithmes du PGCD agissant sur deux entiers ou polynômes. Nous nous orientons maintenant vers des généralisations à la dimension supérieure. Le chapitre 2 présente les résultats obtenus sur les algorithmes de Brun et Knuth qui calculent le PGCD de plusieurs entiers ou polynômes. Nous avons obtenu des résultats *étonnants* en exhibant de nouvelles lois limites (bétas ou uniformes) qui diffèrent des précédentes analyses. Ces travaux ont été réalisés en collaboration avec Brigitte Vallée et Valérie Berthé (laboratoire IRIF, Paris 7). Le chapitre 3 aborde l'analyse des algorithmes de réduction de réseaux euclidiens qui sont une autre généralisation possible des algorithmes du PGCD. Ce travail, issu des travaux de thèse de Mariya Georgieva, est encore en évolution aujourd'hui.

Le chapitre 4 est consacré aux travaux en lien avec la théorie de l'information. Une large part du chapitre décrit les résultats obtenus en collaboration avec Valérie Girardin (laboratoire LMNO, Caen) sur le calcul de taux d'entropie généralisés et renormalisés d'une source. Nous avons obtenu pour la première fois des expressions explicites de taux d'entropie généralisés qui s'appliquent à une large classe de sources qui satisfont une propriété dite des quasi-puissances. La collaboration avec Valérie Girardin a débuté il y a plusieurs années et elle se poursuit encore maintenant avec des travaux en cours. Une petite partie du chapitre est dédiée à un résultat obtenu en collaboration avec Manuel Lladser (Boulder, Colorado) sur la combinatoire des motifs cachés dans un texte produit par une source. Nous avons démontré l'existence d'une loi limite gaussienne pour ce type de motifs lorsque la source est une source dynamique.

Pour terminer, le chapitre 5 décrit mes travaux récents avec Julien David (LIPN, Paris 13), Arnaud Mary (LBBE, Lyon 1) et François Rioult (GREYC, Caen) sur les hypergraphes et leurs traverses minimales. Les traverses minimales d'un hypergraphe correspondent aux motifs de la bordure négative d'une base de données. Ce chapitre est ainsi en lien avec la fouille de données. Nous avons introduit des modèles d'hypergraphes aléatoires avec lesquels nous avons analysé le nombre moyen de traverses minimales. Nous étudions aussi l'algorithme MTMiner qui énumère les traverses minimales d'un hypergraphe.

Publications

L'ensemble des articles est disponible au format `pdf` à l'adresse suivante : `https://lhote.users.greyc.fr/drupal/node/3`. La dernière catégorie de publications rappelle les articles réalisés pendant la thèse.

Revues internationales avec comité de lecture

[1] V. Berthé, L. Lhote, et B. Vallée. Probabilistic analyses of the plain multiple gcd algorithm. *Journal of Symbolic Computation*, 74 :425–474, 2016.

[2] E. Cesaratto, J. Clément, B. Daireaux, L. Lhote, V. Maume-Deschamps, et B. Vallée. Regularity of the Euclid Algorithm ; application to the analysis of fast GCD Algorithms. *Journal of Symbolic Computation*, 44(7) :726 – 767, 2009. International Symposium on Symbolic and Algebraic Computation.

[3] G. Ciuperca, V. Girardin, et L. Lhote. Computation and Estimation of Generalized Entropy Rates for Denumerable Markov Chains. *IEEE Transactions on Information Theory*, 57(7) :4026–4034, 2011.

[4] J. David, L. Lhote, A. Mary, et F. Rioult. An average study of hypergraphs and their minimal transversals. *Theoretical Computer Science*, 596 :124–141, 2015.

[5] V. Girardin et L. Lhote. Rescaling Entropy and Divergence Rates. *IEEE Transactions on Information Theory*, 61(11) :5868–5882, 2015.

[6] L. Lhote et B. Vallée. Gaussian Laws for the Main Parameters of the Euclid Algorithms. *Algorithmica*, 50(4) :497–554, Mar. 2008.

Conférences internationales avec comité de lecture

[7] V. Berthé, J. Creusefond, L. Lhote, et B. Vallée. Multiple GCDs. Probabilistic analysis of the plain algorithm. Dans *Proceedings of the 38th International Symposium on Symbolic and Algebraic Computation*, ISSAC '13, pages 37–44, New York, NY, USA, 2013. ACM.

[8] V. Berthé, L. Lhote, et B. Vallée. Analysis of the Brun Gcd Algorithm. Dans *Proceedings of the ACM on International Symposium on Symbolic and Algebraic Computation*, ISSAC '16, pages 87–94, New York, NY, USA, 2016. ACM.

[9] L. Lhote et M. Lladser. Toward the asymptotic count of bi-modular hidden patterns under probabilistic dynamical sources : a case study. Dans *Discrete Mathematics and Theoretical Computer Science Proceedings*, 23rd International Meeting on Probabilistic, Combinatorial and Asymptotic Methods for the Analysis of Algorithms, AofA 2012, pages 425–452, 2012.

Ateliers internationaux avec comité de lecture

[10] E. Cesaratto, J. Clément, B. Daireaux, L. Lhote, V. Maume-Deschamps, et B. Vallée. Analysis of fast versions of the Euclid Algorithm. Dans *Proceedings of the Fourth Workshop on Analytic Algorithmics and Combinatorics (ANALCO 2007)*, pages 271–285, 2007.

[11] J. Clément, L. Lhote, et B. Vallée. Entropy for dynamical sources. Dans *Proceedings of IWAP 2008 (proceedings sur CD)*, pages 1–10, 2008.

Chapitres de livres

[12] J. Clément et L. Loïck. Analyse d'algorithmes : calculs de PGCD et algorithmes de tri et de recherche. Dans *Informatique Mathématique : Une photographie en 2014, Presses Universitaires de Perpignan*, pages 81–143, 2014.

[13] L. Lhote. Number of frequent patterns in random databases. Dans C. H. Skiadas, editeur, *Advances in Data Analysis (special issue)*, Statistics for Industry and Technology, pages 33–45. Birkhäuser Boston, 2010.

Thèse et publications pendant la thèse

[14] L. Lhote. Computation of a class of continued fraction constants. Dans *Proceedings of the Sixth Workshop on Algorithm Engineering and Experiments and the First Workshop on Analytic Algorithmics and Combinatorics, New Orleans, LA, USA, January 10, 2004*, pages 199–210, 2004.

[15] L. Lhote. *Algorithmes du PGCD et Fouille de Données : le point de vue de l'analyse dynamique*. Thèse, Université de Caen Basse-Normandie, 2006.

[16] L. Lhote, F. Rioult, et A. Soulet. Average number of frequent and closed patterns in random databases. Dans *Proceedings of CAP 05, Conférence francophone sur l'apprentissage automatique, Nice, France*, pages 345–360, 2005.

[17] L. Lhote, F. Rioult, et A. Soulet. Average number of frequent (closed) patterns in bernoulli and markovian databases. Dans *Proceedings of the 5th IEEE International Conference on Data Mining (ICDM 2005), 27-30 November 2005, Houston, Texas, USA*, pages 713–716, 2005.

[18] L. Lhote et B. Vallée. Sharp Estimates for the Main Parameters of the Euclid Algorithm. Dans J. Correa, A. Hevia, et M. Kiwi, editeurs, *LATIN 2006 : Theoretical Informatics*, volume 3887 of *Lecture Notes in Computer Science*, pages 689–702. Springer Berlin Heidelberg, 2006.

Chapitre 1

Analyses probabilistes d'algorithmes : principes et méthodes

L'analyse d'algorithmes est un domaine à l'interface entre l'informatique et les mathématiques qui vise à comprendre le comportement des algorithmes et des structures de données. Le plus souvent, une analyse cherche à déterminer le temps d'exécution ou les ressources mémoires nécessaires à un algorithme pour réaliser ses calculs. Il est aussi possible d'étudier certaines propriétés de l'algorithme (par exemple la qualité de sa réponse) afin de mettre en évidence ses points forts ou ses faiblesses.

L'analyse d'algorithmes a été développée par Knuth dans les années 60. Depuis, le domaine s'est enrichi de nombreuses méthodes et d'axes de recherche relevant des probabilités, des mathématiques discrètes, de la théorie des nombres, de l'algorithmique, etc. Ce mémoire n'a pas pour but d'être exhaustif et nous renvoyons le lecteur à la série de livres de Knuth, *The art of computer programming* [85], pour une introduction très complète à l'analyse d'algorithmes. Dans mes travaux, j'ai souvent utilisé deux méthodes : la combinatoire analytique et l'analyse dynamique. Ce chapitre présente de manière synthétique les deux méthodes. Là encore, nous renvoyons respectivement au livre *Analytic Combinatorics* [60] de Flajolet et Sedgewick et au survey de Vallée [133] pour une présentation de ces approches.

1.1 Objectifs des analyses d'algorithmes

Dans cette section, A désigne un algorithme et $\mathcal{C}$ est l'ensemble des instances sur lequel agit A.

L'objectif d'une analyse est de mesurer le comportement d'un aspect de l'algorithme lorsque la taille de l'entrée augmente. La taille d'un objet est en général liée à l'espace mémoire nécessaire à une machine pour le stocker : la taille d'un polynôme sera son degré, la taille d'un entier sera le nombre de bits pour le représenter, la taille d'un hypergraphe sera son nombre de sommets et d'hyperarêtes, etc. Nous notons $|x|_{\mathcal{C}}$ la taille d'une instance $x \in \mathcal{C}$. L'ensemble des instances de taille n est noté $\mathcal{C}_n$,

$$\mathcal{C}_n = \{x \in \mathcal{C} \mid |x|_{\mathcal{C}} = n\}.$$

Un paramètre X (ou un coût) de l'algorithme A est une application de $\mathcal{C}$ à valeurs réelles ($X : \mathcal{C} \mapsto \mathbb{R}$) qui décrit l'aspect étudié de l'algorithme. Par exemple, X sera le nombre de

divisions euclidiennes des algorithmes du PGCD ou encore le nombre de traverses minimales d'un hypergraphe.

Les analyses dans le pire et le meilleur des cas sont les plus répandues et décrivent les comportements extrémaux de X lorsque la taille des entrées grandit. Ces analyses visent à estimer l'asymptotique des deux suites M_n (meilleur des cas) et P_n (pire des cas) définies par

$$M_n = \inf\{X(\omega) \mid \omega \in \mathcal{C}_n\}, \qquad P_n = \max\{X(\omega) \mid \omega \in \mathcal{C}_n\}.$$

Le pire et le meilleur des cas sont intéressants puisqu'ils donnent des bornes sur le comportement de X. Cependant, ils correspondent souvent à des instances pathologiques de l'algorithme et ne permettent pas de décrire le comportement *générique* de ce dernier.

Une analyse probabiliste ne se focalise plus uniquement sur les cas extrémaux mais attribue un poids à toutes les entrées. Les instances qui apparaissent plus fréquemment dans les applications ont (dans l'idéal) un poids plus important que celles qui apparaissent rarement. Ainsi, les résultats de l'analyse sont plus proches des observations expérimentales et décrivent mieux le comportement *générique* de l'algorithme. Formellement, les poids se modélisent par des distributions de probabilités $\mathbb{P}_n$ sur les ensembles $\mathcal{C}_n$. La valeur moyenne de X sur les instances de taille n est donnée par l'espérance

$$\mathbb{E}_n[X] = \sum_{\omega \in \mathcal{C}_n} \mathbb{P}_n(\omega) X(\omega).$$

L'espérance indique l'ordre de grandeur du paramètre X mais elle ne donne pas d'information sur les variations autour de cet ordre de grandeur. Pour cela, il faut étudier les moments d'ordre supérieur ou la variance définie par

$$\mathbb{V}_n(X) = \mathbb{E}_n[X^2] - \mathbb{E}_n[X]^2.$$

L'inégalité de Bienaymé-Tchebychev fait intervenir la variance à travers l'écart-type $\sigma_n(X) = \sqrt{\mathbb{V}_n(X)}$ et s'écrit

$$\mathbb{P}_n\left[\, |X - \mathbb{E}_n[X]| \geq \alpha \cdot \sigma_n(X)\right] \leq \frac{1}{\alpha^2}, \qquad \alpha > 0. \tag{1.1}$$

L'inégalité montre qu'avec une forte probabilité, X prend ses valeurs autour de sa moyenne dans un intervalle dont la mesure est d'ordre l'écart-type.

Une analyse plus fine encore vise à décrire la répartition des valeurs de X autour de sa moyenne : c'est l'analyse en distribution. Il s'agit de déterminer la fonction f (une densité) dans la limite suivante :

$$\lim_{n\to\infty} \mathbb{P}_n\left[\frac{X - \mathbb{E}_n[X]}{\sigma_n(X)} \leq x\right] = \int_{-\infty}^{x} f(t)dt.$$

C'est un résultat beaucoup plus précis que l'inégalité de Bienaymé-Tchebychev puisque la probabilité de l'équation (1.1) devient une égalité :

$$\mathbb{P}_n\left[\, |X - \mathbb{E}_n[X]| \geq \alpha \cdot \sigma_n(X)\right] = \int_{[t\geq\alpha]} f(t)dt.$$

Les premières analyses des algorithmes du PGCD conduisent souvent à des lois limites gaussiennes dont la densité sur $\mathbb{R}$ est $f(x) = e^{-x^2/2}/\sqrt{2\pi}$. Lors de l'analyse des algorithmes du PGCD, nous avons mis en évidence d'autres lois limites à la fois continues (lois uniformes et bétas) et discrètes (lois géométriques ou quasi-géométriques). Nous reviendrons sur ces lois au chapitre 2.

1.2 Combinatoire analytique

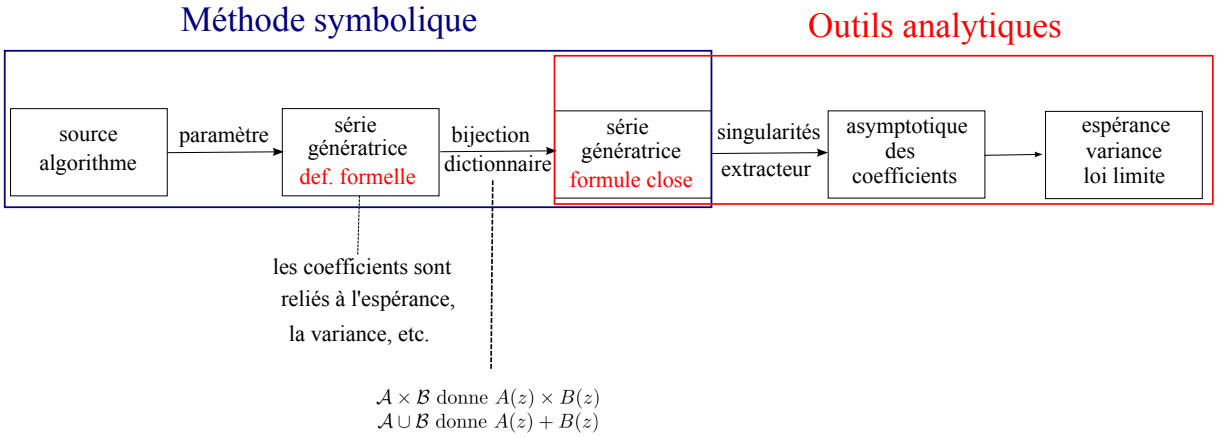
Au cours de mes travaux, j'ai utilisé deux méthodes : la combinatoire analytique et l'analyse dynamique. D'autres approches existent comme la combinatoire algébrique, la combinatoire bijective ou encore les méthodes probabilistes mais nous ne les abordons pas.

Cette partie introduit la combinatoire analytique développée autour de Philippe Flajolet depuis les années 80. Pour une présentation complète, nous renvoyons au livre *Analytic Combinatorics* de Flajolet et Sedgewick [60].

La première section aborde très succinctement les grandes étapes de la combinatoire analytique. Les sections suivantes reprennent avec plus de détails chacune de ces étapes.

1.2.1 Démarche globale

L'idée sous-jacente à la combinatoire analytique est de compter les entrées selon leur taille et la valeur du paramètre étudié. Trouver directement ces quantités est généralement difficile mais la combinatoire analytique les relie aux coefficients de séries génératrices.



La combinatoire analytique se compose de deux grandes phases. La première est la méthode symbolique qui conduit à des manipulations formelles/symboliques de séries génératrices. La seconde considère les séries comme des objets analytiques et s'appuie sur l'analyse complexe.

L'étape symbolique vise à trouver une expression close des séries génératrices en se basant sur des dictionnaires. Intuitivement, si l'ensemble des entrées $\mathcal{C}$ se décompose avec des opérations ensemblistes, alors les dictionnaires transfèrent *automatiquement* la décomposition ensembliste en opérations sur les séries génératrices. Par exemple, si l'ensemble des entrées se décompose en une union disjointe (resp. un produit cartésien, une séquence) d'ensembles, alors la série se décompose en une somme (resp un produit, un quasi-inverse) de séries génératrices *souvent plus simples*. A l'issue de cette première phase, les séries génératrices ont une expression qui peut être exploitée.

Lors de la deuxième phase, les séries génératrices sont vues comme des objets analytiques. L'objectif est d'extraire les coefficients des séries afin de les relier aux quantités probabilistes recherchées. L'analyse complexe montre que l'asymptotique des coefficients d'une série génératrice est liée à la position et à la nature de ses singularités dominantes. Les singularités sont étudiées à travers l'expression des séries génératrices obtenue lors de la méthode symbolique. Des résultats mathématiques que l'on appelle « extracteurs » transfèrent alors *automatiquement* les propriétés des singularités en asymptotiques des coefficients. Ensuite, les asymptotiques suffisent

aux calculs des quantités probabilistes recherchées. Nous détaillons maintenant les étapes de la combinatoire analytique.

1.2.2 Classes combinatoires et séries génératrices

Une classe combinatoire est un ensemble d'objets $\mathcal{C}$ muni d'une taille $|\cdot|_{\mathcal{C}} : \mathcal{C} \mapsto \mathbb{N}$. Bien entendu dans le contexte d'analyse d'un algorithme, la classe combinatoire la plus évidente est celle des entrées (fortuitement déjà appelée $\mathcal{C}$ dans la section précédente). Nous supposons toujours que pour toute taille n donnée, le nombre c_n d'objets de taille n est fini.

La combinatoire analytique vise à estimer l'asymptotique de la suite $(c_n)_n$ en s'appuyant sur les séries génératrices. Il existe plusieurs familles de séries selon que les objets sont *étiquetés* ou pas ou selon les propriétés de la taille (additivité ou multiplicativité). Nous utilisons deux types de séries : les séries ordinaires et les séries de Dirichlet pour des objets non-étiquetés. Les séries ordinaires et de Dirichlet associées à la classe combinatoire $\mathcal{C}$ sont formellement définies par

$$C(z) := \sum_{x\in\mathcal{C}} z^{|x|_{\mathcal{C}}} \quad \text{et} \qquad C(s) := \sum_{x\in\mathcal{C}} \frac{1}{|x|_{\mathcal{C}}^s}. \tag{1.2}$$

En regroupant les objets de même taille, les séries admettent comme expressions alternatives

$$C(z) := \sum_{n\in\mathbb{N}} c_n z^n \quad \text{et} \qquad C(s) := \sum_{n\in\mathbb{N}} \frac{c_n}{n^s}. \tag{1.3}$$

Nous notons $[z^n]C(z)$ ou $[n^{-s}]C(s)$ le coefficient c_n des séries. Les séries $C(z)$ et $C(s)$ sont des séries *univariées* qui sont bien adaptées pour compter les objets de taille fixée. Dans le cadre de l'analyse d'algorithmes, nous sommes aussi intéressés par un coût X lorsque la taille des entrées grandit. Il est donc assez naturel d'ajouter une autre variable qui va « porter » le coût étudié. Les séries ordinaires et de Dirichlet *bivariées* associées à la classe combinatoire $\mathcal{C}$ et au paramètre X (défini sur $\mathcal{C}$ et à valeurs entières) sont formellement définies par

$$C(z,u) := \sum_{\omega\in\mathcal{C}} z^{|\omega|_{\mathcal{C}}} u^{X(\omega)} \quad \text{et} \qquad C(s,u) := \sum_{\omega\in\mathcal{C}} \frac{u^{X(\omega)}}{|\omega|_{\mathcal{C}}^s}. \tag{1.4}$$

En regroupant cette fois-ci les éléments selon leur taille et la valeur de X, les séries s'écrivent sous la forme

$$C(z,u) := \sum_{n\in\mathbb{N}}\sum_{k\in\mathbb{N}} c_{n,k} z^n u^k \quad \text{et} \qquad C(s,u) := \sum_{n\in\mathbb{N}^\star}\sum_{k\in\mathbb{N}} c_{n,k} \frac{u^k}{n^s}, \tag{1.5}$$

avec $c_{n,k}$ le nombre d'objets de la classe $\mathcal{C}$ de taille n pour lesquels X a pour valeur k. Comme précédemment, nous notons $[z^n u^k]C(z,u)$ ou $[n^{-s}u^k]C(s,u)$ le coefficient $c_{n,k}$ des séries.

1.2.3 Probabilités sur les classes combinatoires et séries génératrices

Dans le contexte de l'analyse d'algorithmes, les ensembles $\mathcal{C}_n$ sont munis d'une distribution de probabilité $\mathbb{P}_n$. Si $\mathbb{P}_n$ est la distribution uniforme, calculer des probabilités revient à compter le nombre d'objets à taille et valeur de X fixées. Or c'est exactement ce que représentent les coefficients des séries génératrices. Il suffit donc d'extraire les coefficients des séries pour calculer les probabilités.

La relation suivante relie la série bivariée aux probabilités élémentaires de X :

$$\mathbb{P}_n[X = k] = \frac{c_{n,k}}{c_n} = \frac{[z^n u^k]C(z,u)}{[z^n]C(z)}, \qquad k \in \mathbb{N}. \tag{1.6}$$

Notons que toutes les formules sont similaires avec les séries de Dirichlet. Les moments de X sont aussi accessibles via la dérivation formelle des séries. Par exemple l'espérance vérifie

$$\mathbb{E}_n[X] = \frac{1}{c_n}\sum_{k\in\mathbb{N}} k c_{n,k} = \frac{[z^n]C^{[1]}(z)}{[z^n]C(z)} \quad \text{avec} \quad C^{[i]}(z) = \frac{\partial^i C}{\partial_u^i}(z,u)|_{u=1} \tag{1.7}$$

et pour les moments d'ordres supérieurs, nous avons

$$\mathbb{E}_n[X(X-1)\cdots(X-i)] = \sum_{k\in\mathbb{N}} k(k-1)\cdots(k-i)\frac{c_{n,k}}{c_n} = \frac{[z^n]C^{[i+1]}(z)}{[z^n]C(z)}.$$

La distribution de la variable aléatoire X sur l'ensemble probabilisé $(\mathcal{C}_n, \mathbb{P}_n)$ est entièrement déterminée par sa fonction caractéristique $t \mapsto \mathbb{E}_n[e^{itX}]$ ou sa transformée de Laplace $t \mapsto \mathbb{E}_n[e^{tX}]$. Ce sont deux cas particuliers (avec $u = e^{it}$ ou $u = e^t$) de la série génératrice des moments de X définie par

$$\varphi_n(w) = \mathbb{E}_n[u^X]. \tag{1.8}$$

La série génératrice des moments est fondamentale pour étudier la loi limite du paramètre X. Hwang [79] a par exemple démontré que si $\varphi_n(w)$ ressemble à une puissance, alors X converge vers une loi limite gaussienne. Bien entendu, la série génératrice des moments est reliée à la série génératrice bivariée par la relation

$$\varphi_n(u) = \frac{1}{c_n}\sum_{k\in\mathbb{N}} c_{n,k}u^k = \frac{[z^n]C(z,u)}{[z^n]C(z)}. \tag{1.9}$$

1.2.4 Méthode symbolique : opérations sur les classes et les séries

La précédente section montre le lien entre les séries génératrices et les grandeurs probabilistes du paramètre. Jusqu'à présent, les séries génératrices sont définies formellement mais pour les utiliser, il est nécessaire d'en obtenir une expression alternative *exploitable*. Les expressions alternatives peuvent prendre différentes formes : solutions d'équations différentielles, de systèmes linéaires ou polynomiaux, d'équations fonctionnelles, etc. Dans ce mémoire, les séries génératrices seront souvent des fractions rationnelles explicites ce qui est un cadre très favorable en combinatoire analytique. Une expression explicite est parfois calculable directement depuis la définition avec de la combinatoire simple. La méthode symbolique permet d'aller beaucoup plus loin. Lorsqu'une classe combinatoire *complexe* s'exprime avec des opérations ensemblistes en fonction de classes combinatoires *simples*, alors la méthode symbolique transforme automatiquement les opérations ensemblistes en opérations sur les séries génératrices. Ainsi, la série génératrice de la classe *complexe* s'exprime en fonction des séries génératrices des classes *simples*. Si c'est dernières sont *exploitables*, la série génératrice de la classe *complexe* le devient à son tour.

Nous posons $(\mathcal{A}, |\cdot|_{\mathcal{A}})$ et $(\mathcal{B}, |\cdot|_{\mathcal{B}})$ deux classes combinatoires munies de leur taille respective. Dans la suite, nous utilisons essentiellement trois opérations ensemblistes sur les classes combinatoires :

- l'union disjointe de $\mathcal{A}$ et $\mathcal{B}$, notée $\mathcal{A} \oplus \mathcal{B}$,
- le produit cartésien de $\mathcal{A}$ et $\mathcal{B}$, notée $\mathcal{A} \times \mathcal{B}$

– et l'ensemble des séquences finies sur $\mathcal{A}$, notée $Seq(\mathcal{A})$, définie à l'aide des deux précédents opérateurs par $Seq(\mathcal{A}) = \bigoplus_{n=0}^{\infty} \mathcal{A}^n$.

Il existe d'autres constructions comme les opérateurs *Set* (ensemble), *Cyc* (cycle) ou encore *MSet* (multi-ensemble) mais nous renvoyons à [60] pour de multiples exemples utilisant ces constructions. Pour chaque opération ensembliste sur $\mathcal{A}$ et $\mathcal{B}$, il convient de définir correctement la taille sur le nouvel ensemble formé. La nouvelle taille s'exprime souvent de manière additive ou multiplicative en fonction des tailles de $\mathcal{A}$ et $\mathcal{B}$. Le tableau ci-dessous montre la définition dans les deux contextes.

	Taille additive	Taille multiplicative
Union disjointe $\mathcal{C} = \mathcal{A} \oplus \mathcal{B}$	si $c \in \mathcal{C}$, $\quad \lvert c\rvert_{\mathcal{C}} = \begin{cases} \lvert c\rvert_{\mathcal{A}} & \text{si } c \in \mathcal{A} \\ \lvert c\rvert_{\mathcal{B}} & \text{si } c \in \mathcal{B} \end{cases}$	
Produit cartésien $\mathcal{C} = \mathcal{A} \times \mathcal{B}$	si $c = (a, b)$, $\lvert c\rvert_{\mathcal{C}} = \lvert a\rvert_{\mathcal{A}} + \lvert b\rvert_{\mathcal{B}}$	si $c = (a, b)$, $\lvert c\rvert_{\mathcal{C}} = \lvert a\rvert_{\mathcal{A}} \times \lvert b\rvert_{\mathcal{B}}$
Séquences $\mathcal{C} = Seq(\mathcal{A}) = \bigoplus_{n=0}^{\infty} \mathcal{A}^n$	si $c = (a_1, a_2, \ldots, a_k)$, $\lvert c\rvert_{\mathcal{C}} = \lvert a_1\rvert_{\mathcal{A}} + \lvert a_2\rvert_{\mathcal{A}} + \cdots + \lvert a_k\rvert_{\mathcal{A}}$	si $c = (a_1, a_2, \ldots, a_k)$, $\lvert c\rvert_{\mathcal{C}} = \lvert a_1\rvert_{\mathcal{A}} \times \lvert a_2\rvert_{\mathcal{A}} \times \cdots \times \lvert a_k\rvert_{\mathcal{A}}$

Les propriétés de la taille sont importantes puisqu'elles déterminent le type de séries génératrices à utiliser. L'égalité $z^{|a|_{\mathcal{A}}} z^{|b|_{\mathcal{B}}} = z^{|a|_{\mathcal{A}} + |b|_{\mathcal{B}}}$ indique que les séries génératrices ordinaires sont bien adaptées lorsque la taille est additive. A l'opposé, l'égalité $|a|_{\mathcal{A}}^{-s} |b|_{\mathcal{B}}^{-s} = (|a|_{\mathcal{A}} |b|_{\mathcal{B}})^{-s}$ montre que les séries de Dirichlet sont très utiles lorsque la taille est multiplicative.

La méthode symbolique dispose de dictionnaires qui traduisent les opérations ensemblistes sur les classes combinatoires en opérations sur les séries génératrices. Pour les opérations précédentes, le dictionnaire est donné par le tableau suivant.

	Taille additive Série ordinaire	Taille multiplicative Série de Dirichlet
Union disjointe $\mathcal{C} = \mathcal{A} \oplus \mathcal{B}$	$C(z) = A(z) + B(z)$	$C(s) = A(s) + B(s)$
Produit cartésien $\mathcal{C} = \mathcal{A} \times \mathcal{B}$	$C(z) = A(z) \times B(z)$	$C(s) = A(s) \times B(s)$
Séquences $\mathcal{C} = Seq(\mathcal{A}) = \bigoplus_{n=0}^{\infty} \mathcal{A}^n$	$C(z) = \dfrac{1}{1 - A(z)}$	$C(s) = \dfrac{1}{1 - A(s)}$

Les formules se déduisent directement des définitions formelles des séries et des propriétés additives ou multiplicatives des tailles. On remarque que le dictionnaire est *identique* pour les séries ordinaires ou de Dirichlet. Le dictionnaire s'étend aux séries génératrices bivariées avec exactement les mêmes formules dès que le coût satisfait une propriété d'additivité (à la fois pour les séries ordinaires et de Dirichlet) : pour $c = (a, b)$, $X_{\mathcal{C}}(c) = X_{\mathcal{A}}(a) + X_{\mathcal{B}}(b)$ avec $X_{\mathcal{A}}, X_{\mathcal{B}}, X_{\mathcal{C}}$ des coûts sur les ensembles $\mathcal{A}, \mathcal{B}, \mathcal{C}$.

1.2.5 Les séries génératrices : des objets analytiques

À ce stade de la méthode, les manipulations formelles sont terminées. Les séries génératrices sont maintenant vues comme des fonctions à variables complexes.

Si les séries génératrices ordinaires sont analytiques en 0, les coefficients du développement de Taylor en 0 sont exactement les coefficients de la série génératrice. Les coefficients peuvent se calculer via la formule intégrale de Cauchy,

$$[z^n]C(z) = \frac{1}{2i\pi} \int_{\Gamma} C(z) \frac{dz}{z^{n+1}}, \tag{1.10}$$

avec Γ un chemin simple fermé autour de 0, orienté positivement. La croissance des coefficients d'une série est intimement liée à la nature et la position des singularités dominantes de la série. Une singularité est un point où la fonction cesse d'être bien définie. Pour les séries ordinaires, une singularité dominante est une singularité de plus petit module. La formule de Cauchy combinée avec un contour passant à proximité de la singularité dominante suffit souvent à obtenir l'asymptotique des coefficients. La combinatoire analytique est très riche en résultats qui transfèrent automatiquement les propriétés des singularités dominantes en asymptotiques des coefficients. Dans la suite, nous serons souvent dans un cas très favorable où les séries génératrices ordinaires sont des fractions rationnelles. Un théorème de transfert (ou extracteur) très classique pour les fractions rationnelles est le suivant.

Théorème 1 (Théorème de transfert pour les séries ordinaires)
Si $C(z)$ est de la forme

$$C(z) = \frac{A(z)}{(z-\rho)^\alpha}$$

avec $\alpha \in \mathbb{N}^\star$, $\rho \in \mathbb{C}^\star$ et $A(z)$ analytique sur un disque de rayon plus grand que $|\rho|$ alors

$$[z^n]C(z) \underset{n\to\infty}{\sim} (-1)^\alpha A(\rho)\frac{\rho^{-n-\alpha}}{(\alpha-1)!}n^{\alpha-1}$$

L'analyse complexe permet d'obtenir une asymptotique précise des coefficients avec plusieurs termes. Lors de l'analyse des algorithmes du PGCD, nous utilisons une version plus précise de ce théorème de transfert.

La situation pour les séries de Dirichlet est plus complexe. Les séries de Dirichlet sont analytiques sur un demi-plan (à droite d'une droite verticale). Dans ce cas, une singularité dominante est une singularité sur la droite verticale *au bord* du demi-plan de convergence. L'équivalent de la formule de Cauchy pour les séries de Dirichlet est la formule de Perron.

Théorème 2 (Formule de Perron au premier ordre)
Si $C(s) = \sum_{n\geq 1} c_n n^{-s}$ alors pour tout $N \in \mathbb{R}\backslash\mathbb{N}$,

$$\Phi_n[C] := \sum_{j<n} c_j = \int_\Delta C(s)\frac{n^s}{s}ds$$

avec Δ une droite verticale dans le demi-plan de convergence de $C(s)$.

Contrairement à la formule de Cauchy, c'est la somme cumulée des coefficients qui est obtenue. D'un point de vue combinatoire, cela revient à travailler avec les ensembles cumulés $\widetilde{\mathcal{C}_n} = \cup_{m\leq n}\mathcal{C}_m$ et toutes les formules liant les coefficients des séries génératrices à l'espérance, la variance ou la série génératrice des moments s'adaptent facilement à ce nouveau contexte. L'autre différence majeure avec la formule de Cauchy est le chemin d'intégration. C'est un chemin non borné qui nécessite des hypothèses de convergence uniforme autour de la bande verticale Δ pour être utilisé en pratique. Ces hypothèses sont difficiles à prouver et nous sommes souvent amenés à utiliser des formules de Perron d'ordres supérieurs pour une meilleure convergence.

Comme pour les séries ordinaires, en déplaçant la droite verticale autour de la singularité dominante et en combinant la formule de Perron avec le théorème des résidus, il est possible d'obtenir l'asymptotique de la somme cumulée. Nous obtenons ainsi des théorèmes de transfert pour les séries de Dirichlet, le plus *simple* étant probablement le théorème taubérien de Delange [45].

Théorème 3 (Théorème taubérien de Delange [45])
Soit $C(s) = \sum_{n\geq 1} c_n n^{-s}$ une série à coefficients positifs, analytique dans le demi-plan pointé $\{\Re(s) \geq \sigma\}\backslash\{\sigma\}$ avec $\sigma > 0$, et qui autour de $s = \sigma$ est de la forme

$$C(z) = \frac{A(s)}{(s-\sigma)^{\alpha+1}}$$

avec $\alpha \in \mathbb{N}$ et $A(s)$ analytique en $s = \sigma$. Alors

$$\Phi_n[C] := \sum_{n<N} c_n \underset{N\to\infty}{\sim} \frac{A(\sigma)}{\sigma \cdot \alpha!} N^{\sigma} (\log N)^{\alpha}.$$

Lors de l'analyse des algorithmes du PGCD sur les entiers, nous avons prouvé un nouveau théorème de transfert adapté à notre contexte (voir la section 2.4.5).

Les théorèmes de transfert donnent l'asymptotique des coefficients des séries génératrices. Il suffit d'adapter les formules (1.6), (1.7) ou (1.9) en remplaçant par exemple $[z^n u^k]C(z,u)$ par $[u^k]\Phi_n[C(\cdot,u)]$ pour compléter l'analyse probabiliste du paramètre X.

1.3 Analyse dynamique

Un point clé de la combinatoire analytique est l'utilisation des dictionnaires pour le calcul des séries génératrices. Les dictionnaires s'appuient sur les propriétés additive ou multiplicative de la taille mais ces propriétés n'existent pas toujours. Prenons par exemple l'algorithme d'Euclide. L'opération de base de l'algorithme d'Euclide est la division euclidienne : pour tout couple d'entiers (ou polynômes) (a, b) avec $b \neq 0$, il existe un unique couple d'entiers (ou polynômes) (q, r) avec $0 \leq r < |b|$ (ou $\deg r < \deg b$) tel que $a = bq + r$. La fonction $(a, b) \mapsto (b, q, r)$ définit une bijection. Pour les polynômes, il existe une notion de taille additive, le degré, adaptée à cette bijection :

$$\deg a = \deg b + \deg q.$$

Ainsi, nous utilisons la combinatoire analytique pour l'analyse des algorithmes du PGCD sur les polynômes. En revanche dans le cas des entiers, il n'existe pas de notion de taille additive ni multiplicative adaptée à la division euclidienne, essentiellement à cause de la retenue. La méthode symbolique ne s'applique plus directement.

Au milieu des années 90, Brigitte Vallée a adapté la combinatoire analytique en y ajoutant des outils provenant du domaine des systèmes dynamiques. Cette partie décrit succinctement l'analyse dynamique. Pour une présentation plus complète, nous renvoyons au survey de Brigitte Vallée [133].

1.3.1 Démarche globale

Comme la combinatoire analytique, l'analyse dynamique comporte une étape avec des manipulations formelles sur les séries génératrices et une étape analytique où les séries génératrices sont vues comme des objets analytiques. L'étape analytique est similaire à celle de la combinatoire analytique mais l'analyse dynamique utilise un chemin alternatif équivalent à l'étape formelle.

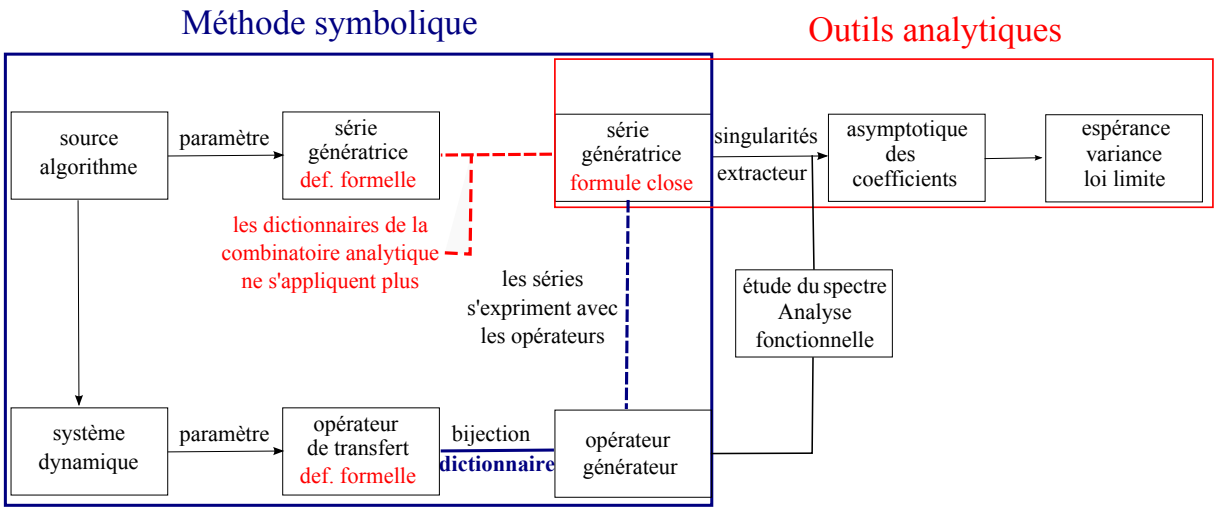


FIGURE 1.1 – Principales étapes d'une analyse dynamique. La méthode symbolique de l'analyse dynamique s'appuie sur les dictionnaires des opérateurs de transfert, les dictionnaires de la combinatoire analytique ne s'appliquant plus.

La première étape d'une analyse dynamique consiste à voir l'algorithme comme un système dynamique, c'est-à-dire une transformation agissant sur un ensemble et que l'on itère. Une itération du système correspond à une itération de l'algorithme et les exécutions de l'algorithme correspondent aux trajectoires finies du système. Par exemple, le système dynamique associé à l'algorithme d'Euclide est le système dynamique des fractions continues aussi appelé transformation de Gauss.

La dynamique de l'algorithme est très liée à la dynamique du système sous-jacent. Des outils classiques pour l'étude de l'évolution d'un système sont les opérateurs de transfert. Le point clé est d'exprimer les séries génératrices en fonction des opérateurs de transfert en s'appuyant comme la combinatoire analytique sur des dictionnaires. Les dictionnaires transforment des opérations ensemblistes (union disjointe, produit cartésien, etc.) en opérations sur les opérateurs (somme d'opérateurs, composition d'opérateurs, etc.). Ainsi, les séries génératrices admettent une formule close en fonction des opérateurs. Ensuite, les opérateurs de transfert et les séries génératrices sont vus comme objets analytiques et les singularités des séries sont liées aux spectres des opérateurs. Cette étape est délicate car la plupart des analyses dynamiques qui ont échoué sont dues à un manque d'informations sur les propriétés spectrales des opérateurs. Toutefois, si l'analyse spectrale est suffisamment précise, la localisation et la nature des singularités est connue et l'utilisation d'extracteurs permet de transférer les propriétés analytiques des séries en asymptotiques sur les coefficients puis en asymptotiques sur les quantités recherchées. Nous détaillons maintenant chacune des étapes d'une analyse dynamique.

1.3.2 Système dynamique

Un système dynamique est une application T qui agit sur un ensemble I. En pratique, l'application T est suffisamment régulière pour être étudiée. Nous supposons toujours qu'il existe une partition topologique $(I_\omega)_{\omega\in\mathcal{A}}$ de I telle que T est injective et de classe au moins $\mathcal{C}^2$ sur chaque ensemble de la partition. L'ensemble $\mathcal{A}$ est appelé l'alphabet du système dynamique.

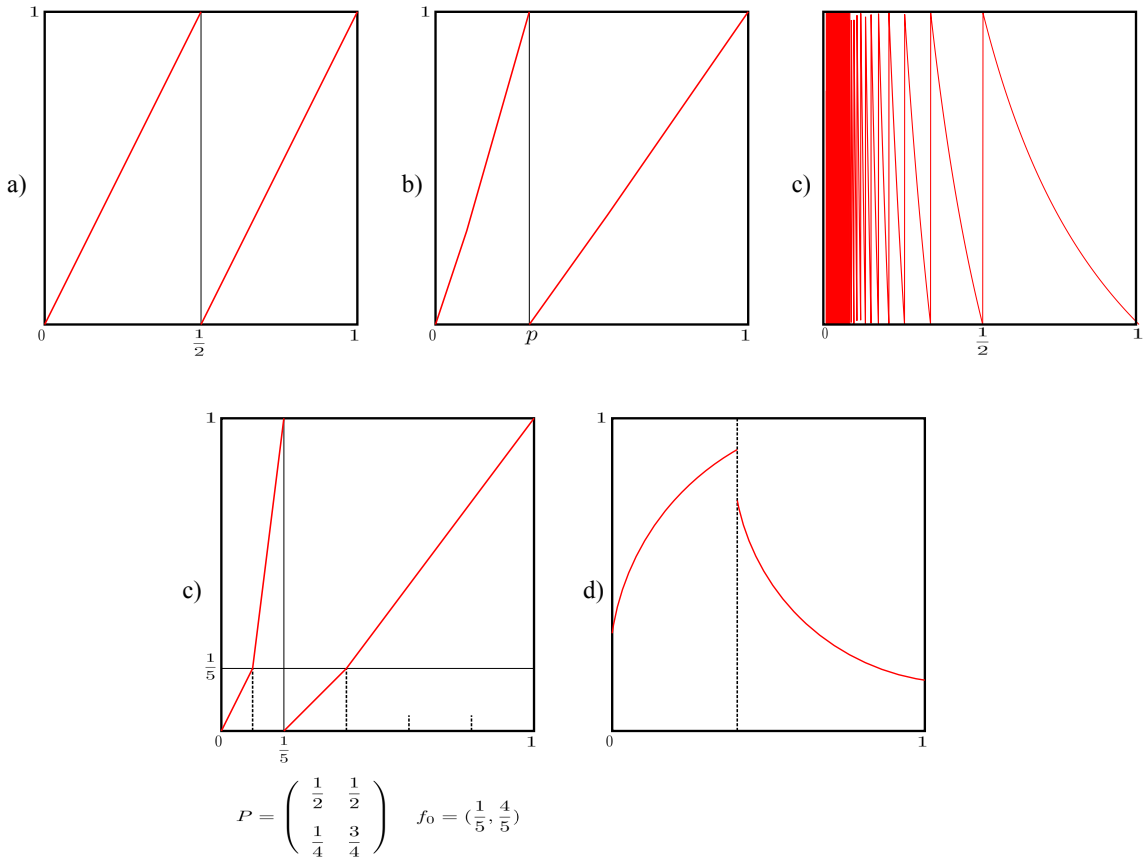


FIGURE 1.2 – Exemples de systèmes dynamiques de l'intervalle avec : a) le système binaire $T(x) = \{2x\}$, b) un système de Bernoulli de paramètre p, $T(x) = \frac{x}{p}\mathbf{1}_{[0,p[}(x) + \frac{x-p}{1-p}\mathbf{1}_{[p,1[}(x)$ c) le système des fractions continues $T(x) = \{1/x\}$, d) une chaîne de Markov de matrice de transition P et de densité initiale f_0, d) un système dynamique général.

Graphiquement si I est un intervalle, l'application T est constituée de branches monotones sur chacun des intervalles de la partition. La figure 1.2 montre plusieurs exemples de systèmes où les branches apparaissent clairement. La restriction de T à chaque intervalle définit une bijection T_ω de I_ω dans $J_\omega := T(I_\omega)$ et la bijection inverse $h_\omega = T_\omega^{-1}$ est appelée branche inverse de profondeur 1. L'ensemble des branches inverses (de profondeur 1) est noté

$$\mathcal{H} = \{h_\omega,\ \omega \in \mathcal{A}\}.$$

Pour un entier $k \in \mathbb{N}$, les branches inverses de profondeur k sont définies par

$$\mathcal{H}^k = \{h_{\omega_1} \circ h_{\omega_2} \circ \cdots \circ h_{\omega_k} |\ \forall i = 1..k, \quad \omega_i \in \mathcal{A}\}$$

et l'ensemble de toutes les branches inverses est noté

$$\mathcal{H}^\star = \bigcup_{k \geq 0} \mathcal{H}^k.$$

Pour un élément $x \in I$, la trajectoire de x, notée $Traj(x)$, est la suite des itérés de x par l'action de T :

$$Traj(x) = (x_0, x_1, x_2, x_3, \dots) \quad \text{avec} \quad x_0 = x, \quad x_i = T(x_{i-1}), \quad i \geq 1.$$

Les branches inverses permettent de relier les points d'une même trajectoire. En effet si $x_k = T^k(x_0) = T_{\omega_k} \circ T_{\omega_{k-1}} \circ \cdots \circ T_{\omega_1}(x_0)$, alors par définition des branches inverses nous avons $x_0 = h_{\omega_1} \circ \cdots \circ h_{\omega_{k-1}} \circ h_{\omega_k}(x_k)$. Lors de l'analyse des algorithmes euclidiens, chaque entrée $\omega \in \mathcal{C}$ correspond à un élément $x \in I$. Une exécution de l'algorithme sur l'entrée ω correspond à une trajectoire finie de x qui se termine (généralement) en 0. Nous pouvons alors écrire $x = h(0)$ avec h une branche inverse. Cette relation est fondamentale pour exprimer les séries génératrices lors de l'analyse des algorithmes euclidiens. Si de plus, les trajectoires possibles

1.3.3 Sources dynamiques et mots

Il est possible de voir un système dynamique comme une source, c'est-à-dire un processus qui produit des mots. Pour cela, il suffit d'une densité initiale sur I, notée f_0 et d'une fonction de codage $\sigma : I \mapsto \mathcal{A}$, constante sur chaque intervalle de la partition et telle que $\sigma(I_\omega) = \omega$ pour tout $\omega \in \mathcal{A}$. Les éléments (I, T, σ, f_0) forment une *source dynamique*.

Pour un élément $x \in I$, le mot infini $M(x)$ associé à x correspond à l'encodage par la fonction σ de la trajectoire de x :

$$M(x) = \sigma(x_0)\sigma(x_1)\sigma(x_2)\sigma(x_3)\dots$$

Pour un mot $\mathbf{w} = \omega_1\omega_2\dots\omega_k$ sur l'alphabet $\mathcal{A}$, l'ensemble des éléments $x \in I$ dont le mot associé commence par $\mathbf{w}$ est noté $I_{\mathbf{w}}$,

$$I_{\mathbf{w}} = \{x \in I \mid M(x) \text{ commence par } \mathbf{w}\}.$$

Notons que $I_{\mathbf{w}}$ peut être vide si aucun mot généré par la source dynamique ne commence par $\mathbf{w}$. Dans le cas contraire, il existe *une unique* branche inverse $h_{\mathbf{w}}$ de profondeur k, définie sur $J_{\mathbf{w}} = T^k(I_{\mathbf{w}})$, telle que $h_{\mathbf{w}}(J_{\mathbf{w}}) = I_{\mathbf{w}}$ et donnée par

$$h_{\mathbf{w}} = h_{\omega_1} \circ h_{\omega_2} \circ \cdots \circ h_{\omega_k}.$$

Autrement dit, il y a une bijection entre les préfixes produits par une source dynamique et les branches inverses. C'est cette bijection qui sera utilisée pour les manipulations formelles en analyse dynamique.

Mais quelle est l'utilité de la densité f_0 ? Une source est un processus probabiliste qui émet des mots. La densité f_0 sert à initialiser le processus. Un élément $x \in I$ est tiré aléatoirement selon la densité f_0. Ensuite, les symboles produits par la source sont ceux du mot $M(x)$. Par exemple pour la source binaire donnée par $T(x) = \{2x\}$ ayant pour codage $\sigma([\frac{i}{2}, \frac{i+1}{2}[) = i$, $i = 1, 2$, $M(x)$ est l'écriture en base 2 de x.

1.3.4 Opérateurs de transfert

Un outil fondamental pour l'étude de la dynamique d'un système est l'opérateur de transfert ou opérateur de Ruelle-Mayer [113]. Ces opérateurs sont similaires aux matrices de transition pour les chaînes de Markov. Ils permettent d'étudier l'évolution de la distribution des données lorsque le système est itéré. Dans le cadre de l'analyse dynamique, ils sont aussi utilisés comme des opérateurs générateurs pour exprimer les séries génératrices.

Pour un mot $\mathbf{w}$ sur l'alphabet $\mathcal{A}$, la probabilité $p_{\mathbf{w}}$ que le système dynamique émette le mot $\mathbf{w}$ est donnée par

$$p_{\mathbf{w}} = \int_{I_{\mathbf{w}}} f_0(t)dt = \int_I \mathbf{G}_{[\mathbf{w}]}[f_0](x)dx, \tag{1.11}$$

avec

$$\mathbf{G}_{[\mathbf{w}]}[f](x) := \mathrm{Jac}[h_{\mathbf{w}}](x) \cdot f \circ h_{\mathbf{w}}(x) \cdot \mathbf{1}_{J_{\mathbf{w}}}(x). \tag{1.12}$$

Ici, l'opérateur $\mathbf{G}_{[\mathbf{w}]}$ fait intervenir le jacobien $\mathrm{Jac}[h_{\mathbf{w}}]$ de $h_{\mathbf{w}}$ et la fonction indicatrice $\mathbf{1}_{J_{\mathbf{w}}}$ de l'ensemble $J_{\mathbf{w}}$. La deuxième égalité s'obtient à l'aide du changement de variable $t = h_{\mathbf{w}}(x)$ avec $x \in J_{\mathbf{w}}$.

Lors de l'analyse des algorithmes euclidiens, le paramètre étudié sera lié à un coût $c : \mathcal{H}^\star \mapsto \mathbb{R}_+$ sur les branches inverses. Il s'avèrera alors très utile de perturber l'opérateur $\mathbf{G}_{[\mathbf{w}]}$ avec deux paramètres complexes s et u de la manière suivante :

$$\mathbf{G}_{[\mathbf{w}],s,u}[f](x) := \mathrm{Jac}[h_{\mathbf{w}}]^s(x) \cdot f \circ h_{\mathbf{w}}(x) \cdot \mathbf{1}_{J_{\mathbf{w}}}(x) \cdot u^{c(h_{\mathbf{w}})}.$$

L'opérateur obtenu s'appelle l'opérateur de transfert pondéré associé à $\mathbf{w}$ et au coût c. L'opérateur de transfert pondéré associé à un ensemble de mots $\mathcal{W}$ se définit par additivité par

$$\mathbf{G}_{[\mathcal{W}],s,u} := \sum_{\mathbf{w}\in\mathcal{W}} \mathbf{G}_{[\mathbf{w}],s,u}.$$

L'opérateur de Ruelle-Mayer est un cas particulier des opérateurs de transfert pondérés. Il correspond à l'ensemble de mots $\mathcal{W} = \mathcal{A}$ (l'alphabet) avec $u = 1$. L'opérateur obtenu s'écrit alors

$$\mathbf{G}_s[f](x) := \sum_{\omega\in\mathcal{A}} \mathrm{Jac}[h_\omega]^s(x) \cdot f \circ h_\omega(x) \cdot \mathbf{1}_{J_\omega}(x).$$

Pour $s = 1$, l'opérateur de Ruelle-Mayer est le transformateur de densité également appelé opérateur de Ruelle-Perron-Frobenius. Sous certaines conditions, il aura les mêmes propriétés de positivité que la matrice de transition d'une chaine de Markov « bien faite » (ergodique, irréductible, apériodique, etc.).

1.3.5 Manipulations formelles des opérateurs

La combinatoire analytique s'appuie sur des dictionnaires qui transfèrent des manipulations sur les classes combinatoires en opérations sur les séries génératrices. L'équivalent pour l'analyse dynamique existe : des opérations ensemblistes sur les branches inverses correspondent à des opérations sur les mots qui correspondent aussi à des opérations sur les opérateurs.

Nous considérons deux ensembles de mots $\mathcal{W}_1$ et $\mathcal{W}_2$ et $\mathcal{W}$ sera le résultat de l'opération sur $\mathcal{W}_1$ et $\mathcal{W}_2$. Les ensembles $\mathcal{H}_1$, $\mathcal{H}_2$ et $\mathcal{H}$ sont les branches inverses respectivement associées aux mots de $\mathcal{W}_1$, $\mathcal{W}_2$ et $\mathcal{W}$. Nous obtenons un dictionnaire pour l'analyse dynamique qui est très similaire à celui obtenu pour la combinatoire analytique.

Mots	Branches inverses	Opérateurs de transfert
Union disjointe $\mathcal{W} = \mathcal{W}_1 \oplus \mathcal{W}_2$	Union disjointe $\mathcal{H} = \mathcal{H}_1 \oplus \mathcal{H}_2$	$\mathbf{G}_{[\mathcal{W}],s,u} =$ $\mathbf{G}_{[\mathcal{W}_1],s,u} + \mathbf{G}_{[\mathcal{W}_2],s,u}$
Concaténation $\mathcal{W} = \mathcal{W}_1 \cdot \mathcal{W}_2$	Composition $\mathcal{H} = \mathcal{H}_1 \circ \mathcal{H}_2$	Composition $\mathbf{G}_{[\mathcal{W}],s,u} =$ $\mathbf{G}_{[\mathcal{W}_2],s,u} \circ \mathbf{G}_{[\mathcal{W}_1],s,u}$
Étoile de Kleene $\mathcal{W} = \mathcal{W}_1^\star = \bigoplus_{n=0}^{\infty} \mathcal{W}_1^n$	$\mathcal{H} = \mathcal{H}_1^\star = \bigoplus_{n=0}^{\infty} \mathcal{H}_1^n$	Quasi-inverse $\mathbf{G}_{[\mathcal{W}],s,u} = (\mathbf{1} - \mathbf{G}_{[\mathcal{W}_1],s,u})^{-1}$.

En théorie des langages, l'étoile de Kleene correspond aux séquences finies de mots. Notez aussi l'inversion de l'ordre des opérateurs lors de l'opération de concaténation de deux ensembles de mots. Comme pour la combinatoire analytique, les opérations sur les opérateurs supposent une propriété d'additivité du paramètre sur les branches inverses : pour $\omega = \omega_1 \cdot \omega_2 \in \mathcal{W}_1 \cdot \mathcal{W}_2$, $c(h_\omega) = c_1(h_{\omega_1}) + c_2(h_{\omega_2})$ avec c, c_1, c_2 des coûts sur les ensembles $\mathcal{H}, \mathcal{H}_1, \mathcal{H}_2$.

1.3.6 Séries génératrices et opérateurs

Le lien entre les séries génératrices et les opérateurs dépend du contexte d'application. Dans le cadre du calcul de PGCD, les algorithmes agissent sur des entrées $\omega = (\omega_0, \omega_1, \ldots, \omega_d)$ et calculent le PGCD de $(\omega_0, \omega_1, \ldots, \omega_d)$. Chaque exécution correspond à une trajectoire unique et finie du système dynamique ou de manière équivalente, à une unique branche inverse h. Nous avons alors les relations fondamentales suivantes (que nous admettons ici),

$$(\frac{\omega_1}{\omega_0}, \ldots, \frac{\omega_d}{\omega_0}) = h(0, \ldots, 0), \qquad \frac{1}{\omega_0^{d+1}} = \mathrm{Jac}[h](0, \ldots, 0),$$

qui sont essentiellement dues au fait que les branches inverses sont des homographies. Nous renvoyons au chapitre 2 pour une explication plus précise de l'origine de ces relations. Si $\mathcal{H}$ est l'ensemble des branches inverses de profondeur 1 et si $\mathcal{H}^\star$ est l'ensemble des branches inverses *construites* par l'algorithme (qui correspondent à des trajectoires finies), alors la série génératrice des entrées ayant un PGCD égal à 1 vérifie

$$\sum_{\omega:\mathrm{PGCD}(\omega)=1} \frac{1}{\omega_0^{(d+1)s}} = \sum_{h\in\mathcal{H}^\star} \mathrm{Jac}[h]^s(0, \ldots, 0) = (\mathbf{1} - \mathbf{G}_{[\mathcal{H}],s,1})^{-1}[\mathbf{1}](0, \ldots, 0).$$

Le quasi-inverse intervient car, selon le dictionnaire, c'est l'opérateur associé à l'ensemble de branches inverses $\mathcal{H}^\star$. Décrire les trajectoires construites par l'algorithme s'avère parfois difficile mais l'approche reste toujours la même. Si l'ensemble des branches inverses est plus complexe, le dictionnaire continue de s'appliquer.

Lors de l'analyse du nombre d'occurrences de motifs particuliers dans un texte aléatoire (voir chapitre 4 section 4.2), la relation fondamentale que nous utilisons est donnée par l'équation (1.11). Cette relation relie la probabilité $p_{\mathbf{w}}$ d'un mot $\mathbf{w} \in \mathcal{A}^\star$ à l'opérateur $\mathbf{G}_{[\mathbf{w}]}$. Par exemple, la série génératrice de tous les mots de $\mathcal{A}^\star$ où la taille d'un mot est son nombre de lettres, est définie par

$$C(z) := \sum_{\mathbf{w}\in\mathcal{A}^\star} p_{\mathbf{w}} z^{|\mathbf{w}|}.$$

L'opérateur associé à l'alphabet $\mathcal{A}$ est $z\mathbf{G}_{[\mathcal{A}]}$ car chaque lettre a pour taille 1 et est *marquée* par la variable z. Selon le dictionnaire, l'opérateur associé à $\mathcal{A}^\star$ est le quasi-inverse $(\mathbf{1} - \mathbf{G}_{[\mathcal{A}]})^{-1}$ mais il faut *marquer* chaque lettre de la variable z. Nous obtenons alors

$$C(z) = \int_I (\mathbf{1} - z\mathbf{G}_{[\mathcal{A}]})^{-1}[f_0](x)dx.$$

Les deux exemples simples ci-dessus résument la manière dont les séries génératrices s'obtiennent à l'aide de manipulations formelles des opérateurs (quitte à adapter un peu le dictionnaire en ajoutant la variable z par exemple). La partie symbolique de l'analyse dynamique est maintenant terminée. Pour résumer, l'algorithme est vu comme un système dynamique et l'évolution de l'algorithme s'étudie à travers la dynamique du système qui se traduit par l'utilisation de branches inverses. Les opérations sur les branches inverses se traduisent en opérations sur les opérateurs et des *relations fondamentales* font le lien avec les séries génératrices.

1.3.7 Les opérateurs : un outil analytique

À ce stade de la méthode, les manipulations formelles des opérateurs ont conduit à une expression des séries génératrices en fonction des opérateurs. Les séries génératrices sont maintenant vues comme des objets analytiques. Les séries s'exprimant avec des opérateurs, nous

sommes ramenés à étudier les propriétés analytiques de ces derniers. Certains opérateurs ont un rôle prédominant : c'est le cas du quasi-inverse $(\mathbf{1} - \mathbf{G}_{[\mathcal{W}],s,u})^{-1}$ qui sera presque toujours l'opérateur qui apportera la singularité dominante. L'analyse fonctionnelle indique que les singularités du quasi-inverse sont intimement liées aux valeurs propres de l'opérateur de transfert associé. Pour une analyse dynamique réussie, il est fondamental que l'opérateur de transfert admette une propriété de type Perron-Frobenius.

Propriété 1 (Perron-Frobenius) *Soit* $\mathbf{G}$ *un opérateur agissant sur un espace (fonctionnel) de Banach* B. *L'opérateur* $\mathbf{G}$ *satisfait la propriété de Perron-Frobenius sur* B *s'il admet une unique valeur propre de module maximum, simple et isolée du reste du spectre par un saut spectral.*

La propriété de Perron-Frobenius est fondamentale : les analyses dynamiques ayant échoué jusqu'à aujourd'hui sont celles où aucun espace fonctionnel B n'a été trouvé.

Soit $\mathbf{G}_s$ un opérateur qui dépend analytiquement d'un paramètre complexe s et qui satisfait la propriété de Perron-Frobenius pour tout s (dans un voisinage complexe fixé). Nous notons $\lambda(s)$ la valeur propre dominante (de module maximum) de $\mathbf{G}_s$ et $\mathbf{Q}_s$ le projecteur sur l'espace propre associé à $\lambda(s)$. Le saut spectral entraine une décomposition spectrale de l'opérateur sous la forme

$$\mathbf{G}_s = \lambda(s)\mathbf{Q}_s + \mathbf{R}_s,$$

avec $\mathbf{R}_s$ l'opérateur associé au reste du spectre et vérifiant $\mathbf{Q}_s \circ \mathbf{R}_s = \mathbf{R}_s \circ \mathbf{Q}_s = 0$. En itérant l'égalité précédente, nous obtenons pour tout entier n

$$\mathbf{G}_s^n = \lambda(s)^n \mathbf{Q}_s + \mathbf{R}_s^n,$$

et en sommant sur n, le quasi-inverse vérifie

$$(\mathbf{1} - \mathbf{G}_s)^{-1} = \frac{\lambda(s)}{1 - \lambda(s)}\mathbf{Q}_s + (\mathbf{1} - \mathbf{R}_s)^{-1}.$$

Nous en déduisons que le quasi-inverse admet un pôle simple lorsque la valeur propre dominante $\lambda(s)$ vaut 1.

Nous savons que pour $(s, u) = (1, 1)$, l'opérateur de transfert $\mathbf{G}_{[\mathcal{A}],1,1}$ est un transformateur de densité. Autrement dit, il transforme une densité en une autre (comme une matrice de transition pour les chaînes de Markov). Sa valeur propre dominante vaut donc 1. Le point $(s, u) = (1, 1)$ constitue une singularité (un pôle simple pour être plus précis) du quasi-inverse. Nous venons de relier les singularités du quasi-inverse aux propriétés spectrales de l'opérateur de transfert.

Une fois les singularités des opérateurs localisées, les singularités des séries génératrices deviennent explicites. La suite de l'analyse dynamique se poursuit comme pour la combinatoire analytique : des théorèmes de transfert sont utilisés pour obtenir les asymptotiques des coefficients qui conduisent au calcul de l'espérance, la variance ou encore à la loi limite du paramètre.

1.4 Conclusion

Dans ce premier chapitre, nous avons introduit les notions fondamentales de l'analyse d'algorithmes. Ensuite, nous avons présenté deux méthodologies que nous utilisons dans ce mémoire : la combinatoire analytique et l'analyse dynamique. Les deux méthodes s'appuient sur les séries génératrices mais diffèrent lors des manipulations formelles. La combinatoire analytique utilise

des dictionnaires qui transfèrent les opérations sur les classes combinatoires en opérations sur les séries génératrices. L'analyse dynamique utilise aussi des dictionnaires mais sur les opérateurs de transfert. Ensuite, les opérateurs sont utilisés pour exprimer les séries génératrices.

Le prochain chapitre est consacré à l'analyse des algorithmes du PGCD et illustrera l'utilisation des deux méthodes.

Chapitre 2

Analyses des algorithmes de calcul du PGCD

Ce chapitre décrit les principaux résultats des articles suivants :
- [1] **Probabilistic analyses of the plain multiple gcd algorithm.** V. Berthé, L. Lhote, B. Vallée.
  Journal of Symbolic Computation, 74 : 425 – 474 , 2016 .
- [8] **Analysis of the Brun GCD algorithm.**
  V. Berthé, L. Lhote, B. Vallée.
  Proceedings of the ACM on International Symposium on Symbolic and Algebraic Computation, ISSAC '16, pages 87–94, New York, USA, 2016.

L'article [7] est une version courte de [1] traitant uniquement du cas des polynômes. L'article [18], rédigé pendant la thèse, est une version courte de [6] que nous évoquons à la fin de ce chapitre avec les articles [10] et [2].

Ce chapitre présente les analyses de trois algorithmes qui calculent le PGCD de deux ou plusieurs entiers/polynômes : les algorithmes d'Euclide, de Knuth et de Brun. J'ai commencé mes travaux sur les algorithmes du PGCD il y a plus de dix ans, pendant ma thèse [15] et ils constituent encore aujourd'hui une part non négligeable de mon travail de recherche. Durant ma thèse, j'ai étudié la loi limite de paramètres non-additifs de l'algorithme d'Euclide comme la complexité binaire ou la taille des entiers à une fraction de l'exécution [18, 6]. Sur la base de ces travaux, nous avons ensuite réalisé la première analyse en moyenne rigoureuse de l'algorithme de Knuth-Schönhage, un algorithme de type *diviser pour régner* [10, 2].

L'analyse dynamique est au cœur de tous ces résultats et a permis de bien comprendre l'analyse des algorithmes agissant sur deux entiers. Il reste des problèmes ouverts mais qui relèvent plus de difficultés techniques que de réels problèmes méthodologiques. Les analyses d'algorithmes du PGCD s'orientent maintenant vers des généralisations de l'algorithme d'Euclide à la dimension supérieure. Plusieurs généralisations existent comme les algorithmes de réduction de réseaux euclidiens (voir chapitre 3) ou encore les algorithmes d'approximations rationnelles simultanées. L'algorithme de Brun fait partie de cette dernière catégorie.

Une des difficultés avec plusieurs entrées est qu'il existe différentes stratégies pour calculer le PGCD : quels éléments choisir pour faire la division euclidienne ? Où placer le résultat de la division ? Doit-on trier ou pas les entrées ? etc. L'algorithme de Knuth adopte une stratégie naïve et effectue des appels successifs à l'algorithme d'Euclide. C'est un algorithme agissant sur plusieurs entrées mais qui reste assez proche de la dimension 2 bien connue. Ceci explique

probablement qu'il fut le premier algorithme *généralisé* que nous ayons analysé. L'algorithme de Brun adopte une stratégie bien différente en effectuant à chaque étape la division euclidienne entre les deux plus grands éléments. C'est un algorithme bien adapté pour les approximations rationnelles simultanées mais nous verrons qu'il est bien moins efficace que l'algorithme de Knuth.

Ce chapitre commence par un court historique des différents travaux sur les algorithmes du PGCD (section 2.1). Ensuite, nous introduisons les algorithmes (section 2.2) et les paramètres étudiés (section 2.3). La section 2.4 présente étape par étape, l'analyse dynamique du nombre d'itérations des algorithmes de Knuth et de Brun. Pour conclure, la section 2.5 revient brièvement sur les résultats d'autres travaux non détaillés dans ce chapitre.

2.1 Contexte

L'algorithme d'Euclide a été décrit en -300 avant notre ère dans le livre *Les Éléments* d'Euclide. Il est considéré par Knuth comme « *le grand-père des algorithmes car c'est le plus vieil algorithme non-trivial encore utilisé aujourd'hui* ». Dès 1845, Lamé réalise une analyse dans le pire des cas de l'algorithme d'Euclide mais c'est seulement autour de 1970 que Heilbronn et Dixon réalisent séparément les premières analyses probabilistes pour des entrées entières [76, 47]. Ils montrent que le nombre de divisions euclidiennes est en moyenne linéaire par rapport à la taille de l'entrée, comme dans le pire des cas mais avec des constantes différentes. En 1994, Hensley [78] effectue la première analyse en distribution et prouve que le nombre d'itérations suit une loi limite gaussienne. Toutefois, la preuve ne s'adapte pas à d'autres paramètres de l'algorithme ni même à d'autres algorithmes. Développée par Brigitte Vallée au milieu des années 90, l'analyse dynamique a permis les analyses de nombreux algorithmes euclidiens ainsi que de nombreux paramètres associés [133]. Parmi les paramètres étudiés, des résultats précis sur le comportement moyen des quotients, de la taille des continuants ou encore du nombre d'itérations ont été obtenus. Akhavi et Vallée ont également analysé la complexité en bits moyenne [27]. En 2002, Viviane Baladi et Brigitte Vallée [29] ont fait une avancée importante. Elles ont étendu la méthode afin d'obtenir les lois limites pour une large classe de coûts dits additifs et à croissance modérée, incluant le nombre d'itérations. Nous nous sommes ensuite appuyés sur leurs travaux pour réaliser l'analyse en distribution de la complexité en bits [18, 6] ou encore pour effectuer l'analyse en moyenne de l'algorithme de Knuth-Schönhage, un algorithme quasi-linéaire de type diviser pour régner [10, 2]. Les analyses dans le cas des polynômes sont plus simples et ne nécessitent pas d'analyse dynamique mais les mêmes résultats ont été obtenus dans [32, 63, 83, 97].

Tous ces travaux montrent que les analyses probabilistes dans le cas de deux entrées (polynômes ou entiers) sont maintenant bien maitrisées même s'il reste des problèmes ouverts. La situation est très différente en dimensions supérieures. Il existe une multitude d'extensions possibles de l'algorithme d'Euclide. L'algorithme le plus naïf pour calculer le PGCD de plusieurs entrées est probablement l'algorithme de Knuth qui présenté dans [85] et qui fait des appels successifs à l'algorithme d'Euclide. Knuth motivait l'utilisation de cet algorithme à l'aide de l'observation suivante : « *Dans la plupart des cas, la taille des PGCD intermédiaires décroît rapidement lors des premiers appels à l'algorithme d'Euclide et cela rend le reste du calcul très rapide* ». La preuve de cette assertion et l'analyse de l'algorithme de Knuth étaient proposées comme un exercice difficile dans la deuxième édition de *The Art of Computer Programming* mais a disparu dans la troisième édition [85]. Nous avons prouvé ces résultats.

L'algorithme d'Euclide est en lien direct avec le développement en fractions continues d'un réel afin d'obtenir ses meilleures approximations rationnelles successives. Plus généralement, les

algorithmes de fractions continues multidimensionnels (et unimodulaires), décrits par exemple dans [120], produisent des approximations diophantiennes de vecteurs réels et donnent tous lieu à des algorithmes calculant le PGCD de plusieurs entiers. La littérature [92, 35] s'intéresse essentiellement à ces approximations diophantiennes dont la qualité dépend fortement des exposants de Lyapunov des systèmes dynamiques sous-jacents. Ces algorithmes trouvent aussi des applications en géométrie discrète [31]. Cependant, les versions discrètes de ces algorithmes qui calculent le PGCD ne sont jamais étudiées. C'est un objectif à long terme de réaliser ces analyses en commençant par l'algorithme de Brun. Notons que ce dernier admet plusieurs descriptions [36, 120] et est aussi connu sous les noms de transformation de Gauss d-dimensionnelle, d'algorithme de Jacobi-Perron ordonné ou encore de « Podsypanin modified Jacobi–Perron algorithm » [74].

2.2 Présentation des algorithmes

Dans la suite, $\mathbb{F}_q[X]$ est l'ensemble des polynômes sur le corps fini $\mathbb{F}_q$ à q éléments. Selon le contexte (polynômes ou entiers), $\mathcal{C}$ désigne soit les entiers non nuls, $\mathcal{C} = \mathbb{N}^\star$, soit les polynômes non nuls à coefficients dans $\mathbb{F}_q$, $\mathcal{C} = \mathbb{F}_q[X]^\star$. Pour un entier x, le module de x est noté $|x|$ et pour un polynôme, $\deg x$ désigne le degré de x.

2.2.1 Algorithme d'Euclide

L'algorithme d'Euclide effectue une succession de divisions euclidiennes pour calculer le PGCD de deux entiers ou polynômes. Pour tout couple $(a, b) \in \mathcal{C} \times \mathcal{C}$, il existe un unique couple d'entiers ou polynômes (q, r), appelés quotient et reste de la division euclidienne de a par b, tel que

$$a = b \cdot q + r, \qquad 0 \leq r < |b| \quad \text{ou} \quad \deg r < \deg b.$$

L'algorithme d'Euclide s'appuie sur la relation $\mathrm{PGCD}(a, b) = \mathrm{PGCD}(b, r)$. Autrement dit, le calcul du PGCD de deux grands entiers/polynômes (a, b) se ramène au calcul du PGCD de deux plus petits entiers/polynômes (b, r). L'algorithme s'arrête lorsque le reste obtenu est nul. Le pseudo-code est donné par l'algorithme 1. Notez la présence de l'échange en début d'algorithme afin d'ordonner les éléments pour les divisions euclidiennes à suivre.

Algorithme 1 : Algorithme d'Euclide

Entrée: Un couple $(a, b) \in \mathcal{C} \times \mathcal{C}$
Sortie: Le PGCD de a et b
  si $|a| < |b|$ (ou $\deg a < \deg b$) **alors**
    échanger a et b
  fin si
  tant que $b \neq 0$ **faire**
    Division euclidienne : $a = b \cdot q + r$ avec $0 \leq r < |b|$ (ou $\deg r < \deg b$)
    $(a, b) \leftarrow (b, r)$
  fin tant que
  retourner a

2.2.2 Algorithme de Knuth

L'algorithme de Knuth calcule le PGCD de $d + 1$ éléments $(u_0, u_1, \ldots, u_d)$ en faisant au plus d appels à un algorithme qui calcule le PGCD de deux éléments. Nous choisissons ici l'algorithme d'Euclide mais nos travaux s'adaptent sans difficulté aux autres algorithmes déjà

étudiés avec l'analyse dynamique. Nous supposons que les éléments $(u_0, u_1, \ldots, u_d)$ appartiennent à l'ensemble

$$\mathcal{C}_{(d)}^{K} = \mathcal{C}^{d+1}.$$

Nous verrons plus loin que l'algorithme de Brun agit sur un ensemble légèrement différent. L'algorithme de Knuth calcule le PGCD g de $(u_0, u_1, \ldots, u_d)$ selon la séquence suivante :

$$g_0 := u_0, \quad g_k = \mathrm{PGCD}(g_{k-1}, u_k) = \mathrm{PGCD}(u_0, u_1, \ldots, u_k), \quad \text{pour } k \in [1, d].$$

Le PGCD $g = g_d$ est obtenu après d phases, chaque phase correspondant à un appel à l'algorithme d'Euclide et faisant disparaitre une composante. En pratique, il est possible d'accélérer l'algorithme en détectant si l'un des PGCDs intermédiaires est trivial (égal à 1 ou de degré nul). Le pseudo-code de l'algorithme *optimisé* est donné par l'algorithme 2. Notez que pour $d = 1$, l'algorithme de Knuth est exactement l'algorithme d'Euclide.

Algorithme 2 : Algorithme de Knuth

Entrée: Un $d+1$-uple $(u_0, u_1, \ldots, u_d) \in \mathcal{C}^{d+1}$
Sortie: Le PGCD de $(u_0, u_1, \ldots, u_d)$.

```
g_0 = u_0
pour i = 1 to d faire
    g_i =Euclide(g_{i-1}, u_i)
    si g_i est trivial (g_i = 1 ou deg g_i = 0) alors
        retourner g_i
    fin si
fin pour
retourner g_d
```

2.2.3 Algorithme de Brun

Il existe plusieurs versions de l'algorithme de Brun. Nous avons choisi une version qui agit sur l'ensemble $\mathcal{C}_{(d)}^{B}$ formé des $(d+1)$-uples d'entiers *non-nuls, distincts* et *ordonnés*,

$$\mathcal{C}_{(d)}^{B} := \{\mathbf{u} = (u_0, u_1, \ldots, u_d) \mid u_0 > u_1 > u_2 > \ldots > u_d > 0\}.$$

Ce choix facilite la description des trajectoires de l'algorithme. A chaque étape, l'algorithme de Brun effectue la division euclidienne entre les deux plus grands éléments (u_0, u_1) et produit un reste r

$$r := u_0 - m u_1, \quad m := \left\lfloor \frac{u_0}{u_1} \right\rfloor,$$

avec $\lfloor x \rfloor$ la partie entière de x. Ensuite, r est inséré dans la liste $\texttt{End}\,\mathbf{u} = (u_1, u_2, \ldots, u_d)$ contenant tous les éléments de $\mathbf{u}$ sans le plus grand u_0. Pour l'insertion, l'algorithme de Brun utilise la procédure $\texttt{InsDis}\,(r, \texttt{End}\,\mathbf{u})$ qui insère r dans la liste ordonnée $\texttt{End}\,\mathbf{u}$ de manière à obtenir une liste d'entiers *non-nuls, distincts* et *ordonnés*. Trois cas peuvent alors se produire :

- (G) (Cas générique) si r n'est pas présent dans $\texttt{End}\,\mathbf{u}$, c'est une insertion classique dans une liste ordonnée.
- (Z) (Cas $r = 0$) Si $r = 0$, *aucune insertion* n'est réalisée. La procédure retourne $\texttt{End}\,\mathbf{u}$ et le nombre de composantes diminue de 1.
- (E) (Cas d'égalité) Si $r \neq 0$ est déjà présent dans $\texttt{End}\,\mathbf{u}$, *aucune insertion* n'est réalisée. La procédure retourne $\texttt{End}\,\mathbf{u}$ et le nombre de composantes diminue de 1.

L'algorithme se décompose donc en d phases, numérotées de 1 à d, qui correspondent chacune à la *disparition* d'une composante. Pendant la phase i, l'algorithme agit sur l'ensemble $\mathcal{C}^B_{(d-i+1)}$ et s'arrête sur un élément de $\mathcal{C}^B_{(d-i)}$ ayant une composante de moins. Lorsqu'il n'en reste plus qu'une, l'algorithme se termine et la dernière composante correspond au PGCD. Pour résumer, chaque étape de l'algorithme de Brun pendant la phase $i \in [1, d]$ se décrit par la transformation

$$U_{(d-i+1)}(\mathbf{u}) = \texttt{InsDis}\,(u_0 \bmod u_1, \texttt{End}\,\mathbf{u})\,. \tag{2.1}$$

Le pseudo-code de l'algorithme de Brun est donné par l'algorithme 3.

Algorithme 3 : Algorithme de Brun

Entrée: Un $d+1$-uple $\mathbf{u} = (u_0, u_1, \ldots, u_d)$ d'entiers de $\mathcal{C}^B_{(d)}$
Sortie: Le PGCD de $(u_0, u_1, \ldots, u_d)$.
{on effectue d phases numérotées de 1 à d}
pour $i = 1$ **à** d **faire**
tant que $\mathbf{u} \in \mathcal{C}^B_{(d-i+1)}$ **faire**
$\mathbf{u} = U_{(d-i+1)}(\mathbf{u}) = \texttt{InsDis}\,(u_0 \bmod u_1, \texttt{End}\,\mathbf{u})$
fin tant que
fin pour
retourner $\mathbf{u}$

2.3 Paramètres étudiés et résultats

Notre objectif est d'analyser aussi précisément que possible différents aspects des algorithmes afin de mieux comprendre leur comportement et de pouvoir les comparer. Nous commençons par introduire les paramètres d'intérêt des algorithmes et les modèles aléatoires avant de présenter les principaux résultats.

2.3.1 Paramètres des algorithmes

Au cours de mes travaux, j'ai eu l'occasion de m'intéresser à de nombreux paramètres des algorithmes du PGCD. Les paramètres peuvent être regroupés en trois grandes familles. Il y a tout d'abord les paramètres dits additifs qui s'expriment comme la somme d'un coût élémentaire sur les quotients apparaissant lors d'une exécution. C'est le cas du nombre d'itérations où le coût élémentaire vaut 1. Nous donnerons d'autres exemples plus tard.

Les paramètres décrivant l'évolution des entiers/polynômes au cours de l'algorithme, aussi appelés continuants, forment une autre famille de coûts. Par exemple dans [18] , nous avons étudié la taille binaire des entiers à une fraction de l'exécution et nous avons montré que cette taille décroit régulièrement au fur et à mesure que l'algorithme d'Euclide s'exécute. Dans [1, 8, 7], nous avons analysé la taille des PGCD intermédiaires au début de chaque phase et deux types de lois limites ont été mis en évidence.

Pour terminer, il y a les coûts mixtes qui mélangent multiplicativement les deux précédentes familles. La complexité binaire de l'algorithme d'Euclide mesure le nombre d'opérations sur les bits et constitue un coût mixte.

Afin de simplifier la lecture de ce mémoire, j'ai décidé de ne présenter que les résultats liés aux paramètres additifs. Les autres résultats seront évoqués mais avec une précision moindre. Dans toute la suite, si X désigne un paramètre, X^B et X^K désigne le même paramètre mais

appliqué respectivement à l'algorithme de Brun et de Knuth. Nous utiliserons la notation X uniquement lorsque les explications s'appliquent aux deux algorithmes.

2.3.1.1 Paramètres additifs

Pour une entrée $\mathbf{u} = (u_0, u_1, \ldots, u_d)$, nous notons $L(\mathbf{u})$ le nombre d'itérations (de divisions euclidiennes) effectuées par l'algorithme pour calculer le PGCD. Lors de la i-ème itération ($i \in [1, L(\mathbf{u})]$), l'algorithme réalise la division euclidienne entre deux éléments (polynômes ou entiers) $a_i(\mathbf{u})$ et $b_i(\mathbf{u})$. Le quotient et le reste obtenus sont respectivement notés $q_i(\mathbf{u})$ et $r_i(\mathbf{u})$. Un paramètre X est dit additif s'il est de la forme

$$X(\mathbf{u}) = \sum_{i=1}^{L(\mathbf{u})} c(q_i(\mathbf{u}))$$

avec c un coût élémentaire positif sur les quotients ($c : \mathcal{C} \mapsto \mathbb{R}^+$). Lors des analyses des algorithmes de Brun et de Knuth, nous nous sommes intéressés à trois coûts additifs.

Nombre d'itérations pendant une phase : L_k
Une exécution des algorithmes de Brun et de Knuth agissant sur $d+1$ entrées $\mathbf{u} = (u_0, u_1, \ldots, u_d)$ se décompose en exactement d phases, numérotées de 1 (la première) à d (la dernière), chacune faisant « disparaître » une composante. Pendant chaque phase, les algorithmes effectuent des divisions euclidiennes. Pour $k \in [1, d]$, le paramètre L_k compte le nombre de divisions euclidiennes réalisées pendant la phase k de l'algorithme. Il correspond à un coût élémentaire c défini par

$$c(q_i(\mathbf{u})) = 1 \text{ si } q_i(\mathbf{u}) \text{ est un quotient de la phase } k, \text{ 0 sinon.}$$

Nombre total d'itérations : L
Le nombre total d'itérations des algorithmes est juste la somme des itérations de chaque phase. Autrement dit pour des algorithmes agissant sur $d+1$ entrées, nous avons $L = L_1 + L_2 + L_3 + \cdots + L_d$. Il correspond à un coût élémentaire constant $c \equiv 1$.

Nombre de quotients égaux à 1 dans la phase 1 : M_1
Lorsque qu'un quotient vaut 1, la division euclidienne est en fait une soustraction. Il existe une version soustractive de l'algorithme de Brun qui n'utilise que des soustractions. Le paramètre M_1 permet de voir si la version multiplicative de l'algorithme (avec des divisions) est relativement proche de la version soustractive. Le paramètre M_1 correspond à un coût élémentaire c défini par

$$c(q_i(\mathbf{u})) = 1 \text{ si } q_i(\mathbf{u}) = 1 \text{ et } q_i(\mathbf{u}) \text{ est un quotient de la phase 1, 0 sinon.}$$

2.3.2 Notions de tailles et modèles probabilistes

Avant d'énoncer les résultats, il est nécessaire de définir une notion de taille. Les analyses des algorithmes de Brun et de Knuth ont été réalisées avec deux notions de taille différentes.

Taille pour l'algorithme de Knuth :
La taille utilisée pour les analyses de l'algorithme de Knuth est proche de la taille totale nécessaire pour stocker le vecteur d'entrée. Précisément pour $\mathbf{u} = (u_0, u_1, \ldots, u_d) \in \mathcal{C}_{(d)}^K$, la taille de $\mathbf{u}$ est donnée par

$$|\mathbf{u}|_K = \begin{cases} \deg u_0 + \deg u_1 + \cdots + \deg u_d & \text{dans le cas des polynômes,} \\ \lfloor \ln u_0 + \ln u_1 + \cdots + \ln u_d \rfloor & \text{dans le cas des entiers.} \end{cases}$$

Ces notions de tailles ont été choisies car elles se manipulent facilement avec les séries génératrices. Pour un entier n, l'ensemble des entrées de taille n de l'algorithme de Knuth est alors défini par

$$\mathcal{C}^K_{(d),n} = \{\mathbf{u} \in \mathcal{C}^K_{(d)} \mid |\mathbf{u}|_K = n\}.$$

Nous munissons les ensembles $\mathcal{C}^K_{(d),n}$ de la distribution uniforme.

Taille pour l'algorithme de Brun :
Contrairement à l'algorithme de Knuth, la taille utilisée pour les analyses de l'algorithme de Brun est donnée par la taille maximum des composantes de l'entrée. Précisément pour $\mathbf{u} = (u_0, u_1, \ldots, u_d) \in \mathcal{C}^B_{(d)}$, la taille de $\mathbf{u}$ est donnée par

$$|\mathbf{u}|_B = \lfloor \ln u_0 \rfloor.$$

Nous nous limitons à des entrées entières pour l'algorithme de Brun. Pour un entier n, l'ensemble des entrées de taille n est alors défini par

$$\mathcal{C}^B_{(d),n} = \{\mathbf{u} \in \mathcal{C}^B_{(d)} \mid |\mathbf{u}|_B = n\}.$$

Nous munissons les ensembles $\mathcal{C}^B_{(d),n}$ de la distribution uniforme.

Dans la suite, la mention *taille totale* fera référence à la taille $|\cdot|_K$ utilisée pour l'algorithme de Knuth alors que nous parlerons de *taille max* pour pour la taille $|\cdot|_B$ utilisée pour l'algorithme de Brun.

Comparaison des deux modèles aléatoires :
Afin de comparer les algorithmes, il aurait été idéal d'utiliser la même notion de taille et donc les mêmes modèles probabilistes. Cependant, les deux modèles sont assez différents. Dans le modèle avec la taille max, un élément aléatoire de taille n a très probablement des composantes qui sont toutes de tailles (nombre de bits ou degré) très proches de n. A contrario dans le modèle avec la taille totale, une entrée comportant à la fois des petites et des grandes composantes est aussi probable qu'une entrée ayant des composantes de tailles similaires.

Les deux modèles probabilistes étant assez différents, il est difficile de comparer les résultats et de fait, de comparer les deux algorithmes. Nous nous y risquerons tout de même mais avec la plus grande prudence...

2.3.3 Principaux résultats

Les modèles probabilistes et les paramètres d'intérêt étant définis, nous pouvons maintenant énoncer les principaux résultats obtenus. Nous avons obtenu des résultats fins pour les algorithmes d'Euclide et de Knuth puisque nous avons réalisé des analyses en distribution en déterminant plusieurs lois limites. Faute d'informations précises sur les propriétés analytiques des opérateurs, les analyses pour l'algorithme de Brun se limitent pour l'instant à des analyses en moyenne. Nous donnons plus de détails dans la section 2.4.

2.3.3.1 Analyses en moyenne pour l'algorithme de Knuth

Nous avons étudié l'algorithme de Knuth à la fois pour des entrées entières et polynomiales. Dans les deux cas, les résultats sont très similaires. Les analyses en moyenne montrent que l'algorithme de Knuth effectue l'essentiel de son travail lors de la première phase. Précisément,

l'algorithme effectue en moyenne un nombre linéaire de divisions pendant la première phase alors qu'il en effectue un nombre constant pour les autres phases. Les résultats sur les entiers font intervenir plusieurs objets liés au système dynamique de Gauss (système dynamique de l'algorithme d'Euclide) dont la transformation est donnée par $T(x) = 1/x - \lfloor 1/x \rfloor$ sur l'intervalle $]0, 1]$ et $T(0) = 0$. L'opérateur de transfert associé est défini par

$$\mathbf{H}_s[f](x) = \sum_{m \geq 1} \frac{1}{(m+x)^s} f\left(\frac{1}{m+x}\right). \tag{2.2}$$

Sur un espace fonctionnel adéquat (par exemple $C^1([0,1])$), il vérifie la propriété de Perron-Frobenius. Il admet pour $s > 1$ une unique valeur propre de module maximum $\lambda(s)$. Nous introduisons aussi la fonction $\varphi_{s,t}$ définie pour $\Re s > 1$ et $\Re t > 1$ comme

$$\varphi_{s,t}(x) := \frac{1}{2} \frac{\zeta(s+t)}{\zeta(s)\zeta(t)} (I - \mathbf{H}_{s+t})^{-1} \circ (\mathbf{H}_s + \mathbf{H}_t)[1](x)\,, \tag{2.3}$$

avec $\zeta(s)$ la fonction zêta de Riemann. On peut étendre la fonction en $t = 1$ (ou $s = 1$ par symétrie) et en $(s,t) = (1,1)$ par

$$\varphi_{s,1}(x) = \frac{1}{2} \frac{\zeta(s+1)}{\zeta(s)} (\mathbf{1} - \mathbf{H}_{s+1})^{-1}[\mathbf{1}](x), \qquad \varphi_{1,1}(x) = \frac{1}{1+x}. \tag{2.4}$$

Le théorème suivant résume les résultats d'analyse en moyenne des paramètres additifs pour l'algorithme de Knuth.

Théorème 4 [Nombre moyen d'itérations pendant les phases] *Considérons $\mathcal{C}^K_{(d),n}$ l'ensemble des $(d+1)$-uplets de taille totale n muni de la distribution uniforme. Alors, le nombre d'itérations pendant les phases de l'algorithme de Knuth vérifie les propriétés probabilistes suivantes :*

(*a*) *Le nombre moyen d'itérations L_1 pendant la première phase est linéaire en la taille n des entrées et vérifie*

$$\mathbb{E}_n[L_1] = \frac{1}{\mathcal{E}_1} \frac{n}{d+1} + O(1)\,,$$

où $\mathcal{E}_1$ est l'entropie du système dynamique de l'algorithme d'Euclide. En particulier, $\mathcal{E}_1 = \frac{2q}{q-1}$ pour les polynômes et pour les entiers $\mathcal{E}_1 = \frac{\pi^2}{6 \ln 2}$.

(*b*) *Pour tout $k \in [2..d]$, le nombre moyen d'itérations L_k pendant la k-ème phase est asymptotiquement constant et vérifie*

$$\mathbb{E}_n[L_k] = a_k + o(1)\,,$$

avec $a_k = \dfrac{q^k - 1}{q^k - q}$ pour les polynômes et dans le cas des entiers,

$$a_k = (I - \mathbf{H}_{k+1})^{-1}[\varphi_{k,1}](0).$$

2.3.3.2 Analyses en moyenne pour l'algorithme de Brun

Contrairement à l'algorithme de Knuth, les analyses de l'algorithme de Brun ne concernent que des entrées entières et non les polynômes. La difficulté intervient lorsque des polynômes de même degré apparaissent après une division. Il faut alors choisir où positionner le reste de

la division et ces choix compliquent la description des trajectoires du système dynamique sous-jacent. Décrire les trajectoires est une étape fondamentale de l'analyse dynamique, c'est pourquoi nous nous sommes limités au cas d'entrées entières.

Comme pour l'algorithme de Knuth, nous avons étudié le nombre d'itérations de chaque phase de l'algorithme de Brun. Nous avons aussi souhaité comparer l'algorithme de Brun avec sa version soustractive qui n'effectue que des soustractions. Les résultats obtenus sont résumés dans le théorème suivant.

Théorème 5 *Considérons l'ensemble $\mathcal{C}^B_{(d),n}$ des $(d+1)$-uplets de taille-max n (i.e. $\lfloor \ln u_0 \rfloor = n$) muni de la distribution uniforme. Alors, l'algorithme de Brun vérifie les propriétés probabilistes suivantes :*

(*a*) *Le nombre total de divisions L et le nombre L_1 de divisions effectuées lors de la première phase vérifient*

$$\mathbb{E}_n[L] \sim \mathbb{E}_n[L_1] \sim \frac{d+1}{\mathcal{E}_d} \cdot n,$$

où $\mathcal{E}_d$ est l'entropie du système dynamique de Brun en dimension d.
De plus, lorsque d tend vers l'infini, l'entropie $\mathcal{E}_d$ vérifie $\mathcal{E}_d \sim \ln d$.

(*b*) *Le nombre $L_2 + L_3 + \cdots + L_d$ d'itérations effectuées pendant le reste de l'exécution (après la première phase) est en moyenne asymptotiquement constant.*

(*c*) *Soit M_1 le nombre de quotients égaux à 1 pendant la première phase. Le rapport entre les espérances $\mathbb{E}_n[M_1]$ et $\mathbb{E}_n[L_1]$ tend vers une constante $\rho_d < 1$. Lorsque $d \to \infty$, la constante ρ_d tend vers 1 avec une vitesse de convergence en $O(2^{-d/\log d})$.*

Les résultats du théorème 5 sont très similaires à ceux obtenus au théorème 4 pour l'algorithme de Knuth. Dans les deux cas, le nombre total d'itérations est linéaire en la taille et la quasi-totalité des divisions sont effectuées pendant la première phase. Pourtant, la dimension a un rôle différent dans la complexité des deux algorithmes.

Le théorème 4 indique que le nombre moyen d'itérations de l'algorithme de Knuth est d'ordre $n/((d+1)\mathcal{E}_1)$. Or dans le modèle de Knuth, $n/(d+1)$ est la taille moyenne de chaque entrée dont u_0. De manière très grossière, la complexité moyenne de l'algorithme de Knuth est d'ordre $(1/\mathcal{E}_1) \times \mathbb{E}_n[\ln u_0]$ alors qu'elle est d'ordre $(d+1)\mathbb{E}_n[\ln u_0]/\mathcal{E}_d$ pour l'algorithme de Brun. Comme $\mathcal{E}_d \sim \ln d$, le nombre d'itérations de l'algorithme de Brun est sous-linéaire en la dimension alors qu'il est indépendant de la dimension pour l'algorithme de Knuth. Nous avons réalisé des expériences afin de comparer les deux algorithmes en fonction de la dimension d et lorsque la taille-max (au sens de Brun) est fixée. La figure 2.1 montre que notre intuition est correcte.

Nous avons aussi étudié le pire des cas de l'algorithme de Brun qui, à notre connaissance, n'était pas connu.

Théorème 6 *Pour tout entier positif d, notons $\tau_d \in]0,1[$ la plus petite racine non nulle du polynôme $z^{d+1} + z - 1$. Pour tout entier n, le nombre maximum d'itérations de l'algorithme de Brun $Q_{d,n}$ sur l'ensemble $\mathcal{C}^B_{(d),n}$ vérifie,*

$$Q_{d,n} \sim \frac{1}{|\ln \tau_d|} n \qquad (n \to \infty).$$

De plus lorsque $d \to \infty$, le réel τ_d vérifie $1/|\ln \tau_d| \sim (d+1)/\ln d$.

La preuve du théorème 6 met en évidence que le pire des cas se produit lorsque tous les quotients ont pour valeur 1, autrement dit, lorsqu'il n'y a que des soustractions. L'asymptotique dans le

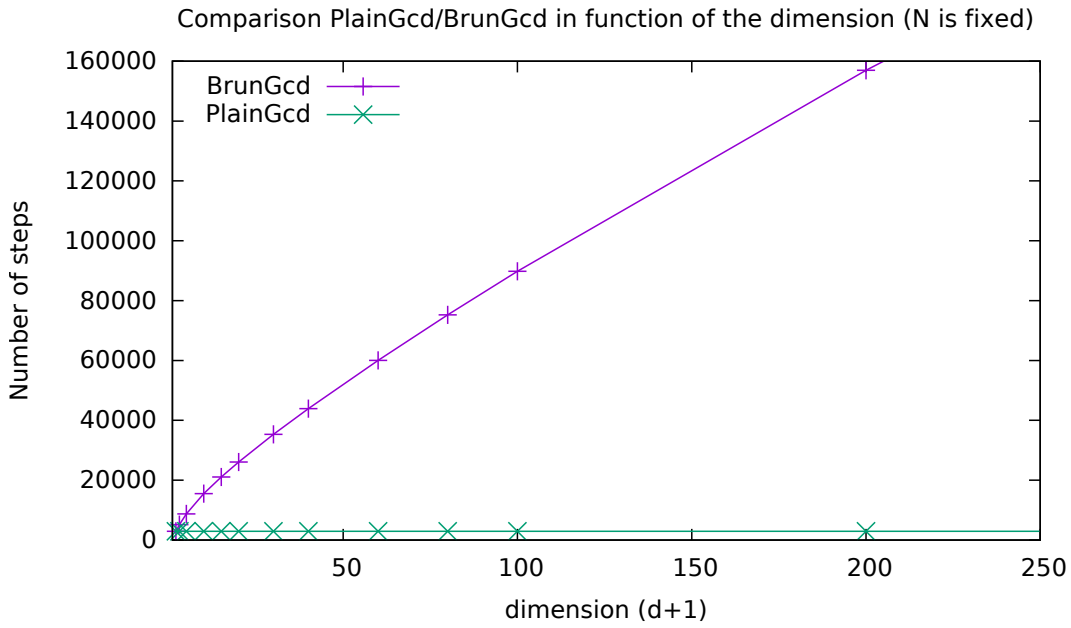


FIGURE 2.1 – Nombre d'itérations de l'algorithme de Brun (`BrunGcd`) et de l'algorithme de Knuth (`PlainGcd`) pendant la première phase en fonction de d. Les entiers utilisés ont au plus $N = 5000$ bits.

pire des cas fait intervenir le rapport $(d+1)/\ln d$ qui est également la constante dans le cas moyen. Comme le pire des cas est atteint lorsque tous les quotients ont pour valeur 1, cela semble indiquer que l'algorithme de Brun fait intervenir des quotients qui sont presque toujours égaux à 1. C'est ce qu'indique le (c) du théorème 5. Nous avons aussi réalisé des expériences pour illustrer ce phénomène. La figure 2.2 montre que la proportion de quotients égaux à 1 pendant la première phase tend rapidement vers 1 lorsque la dimension augmente. Lorsque $d = 16$, plus de 99% des divisions sont en fait des soustractions et pour $d = 50$, cette proportion est 99.99%.

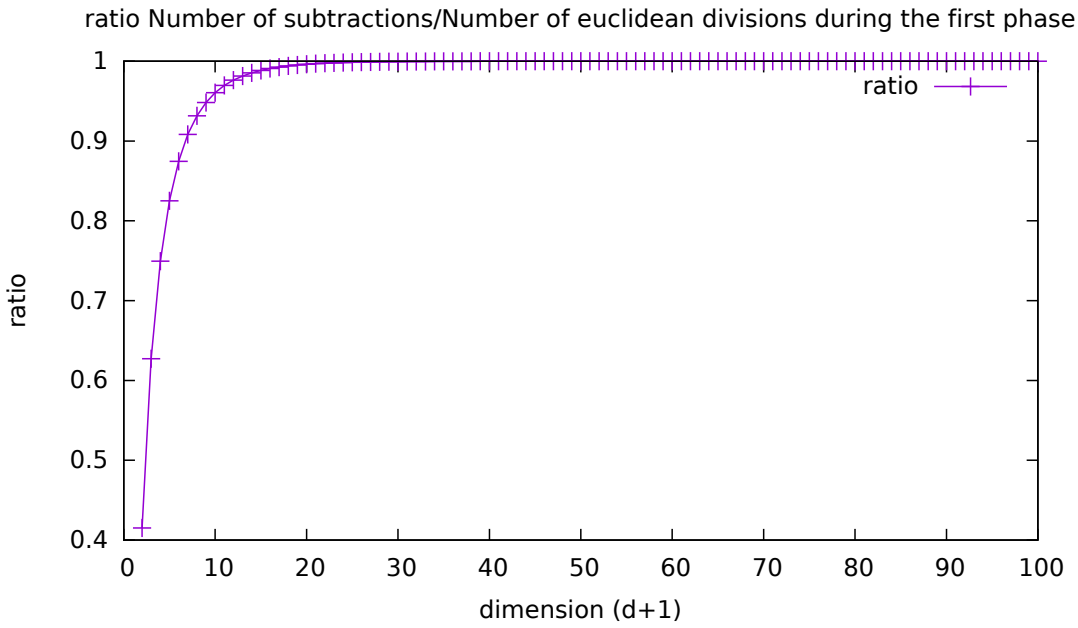


FIGURE 2.2 – Proportion de quotients égaux à 1 lors de la première phase de l'algorithme de Brun en fonction de la dimension. Les entiers utilisés ont au plus $N = 5000$ bits.

2.3.3.3 Analyses en distribution pour l'algorithme de Knuth

L'algorithme de Knuth s'appuie sur l'algorithme d'Euclide dont l'analyse est bien maitrisée puisque nous disposons de nombreux résultats sur le système dynamique sous-jacent. En particulier, les propriétés spectrales et analytiques des opérateurs de transfert sont bien connues. Nous avons exploité ces propriétés pour réaliser des analyses en distribution. Les résultats obtenus sur les paramètres additifs sont résumés par le théorème suivant.

Théorème 7 [Loi limite des itérations pendant les phases] *Considérons $\mathcal{C}^K_{(d),n}$ l'ensemble des $(d+1)$-uplets de taille totale n muni de la distribution uniforme. Alors, le nombre d'itérations pendant les phases de l'algorithme de Knuth vérifient les propriétés probabilistes suivantes :*

(*a*) *Le nombre d'itérations L_1 pendant la première phase suit asymptotiquement une loi béta de paramètre $(1, d)$ sur l'intervalle $[0, 1/\mathcal{E}_1]$. Précisément, la probabilité $\mathbb{P}_n[L_1 > m]$ de l'évènement $[L_1 > m]$ vérifie lorsque $n \to \infty$ et $m/n \in [0, 1/\mathcal{E}_1]$*

$$\mathbb{P}_n[L_1 > m] = \left(1 - \mathcal{E}_1 \frac{m}{n}\right)^d + O\left(\varepsilon_n\right)$$

avec ε_n un terme d'erreur explicite et uniforme lorsque $m/n \in [0, 1/\mathcal{E}_1]$ ($\varepsilon_n \to 0$).

(*b*) *Pour $k \geq 2$, le nombre d'itérations L_k pendant la k-ième phase suit asymptotiquement une loi géométrique dans le cas polynômial. La probabilité $\mathbb{P}_n[L_k > m]$ de l'évènement $[L_k > m]$ vérifie lorsque $n \to \infty$ et $m/n \in \left[0, \frac{1}{k+1}\frac{q^k-1}{q^k}\right]$*

$$\mathbb{P}_n[L_k > m] = \left(\frac{q-1}{q^k-1}\right)^m + o\left(1\right),$$

où le terme d'erreur $o(1)$ est explicite et uniforme dans l'intervalle fermé $\left[0, \frac{1}{k+1}\frac{q^k-1}{q^k}\right]$.

(*c*) *Pour $k \geq 2$, le nombre d'itérations L_k pendant la k-ième phase suit asymptotiquement une loi dite quasi-géométrique dans le cas entier. La probabilité $\mathbb{P}_n[L_k > m]$ de l'évènement $[L_k > m]$ vérifie lorsque $n \to \infty$ et $m/n \in [0, \lambda(k+1)/|\lambda'(k+1)|]$,*

$$\begin{aligned} \mathbb{P}_n[L_k > m] &= \mathbf{H}_{k+1}^m[\varphi_{k,1}](0) + O(\varepsilon_n) \\ &= b_k \lambda(k+1)^m + O(\rho_k^m + \varepsilon_n) \end{aligned}$$

avec ε_n un terme d'erreur explicite et uniforme lorsque $m/n \in [0, \lambda(k+1)/|\lambda'(k+1)|]$, $0 < \rho_k < \lambda(k+1)$ et b_k une constante qui ne dépend que de $\mathbf{H}_{k+1}$ et $\varphi_{k,1}$.

(*d*) *Le nombre total de divisions euclidiennes L admet la même asymptotique que le paramètre L_1. En particulier, L a asymptotiquement la même espérance que L_1 donnée au théorème 4-(a) et suit la même loi limite béta donnée au (c) ci-dessus.*

La figure 2.3 montre les lois limites décrites dans le théorème 7. Chaque graphique est le résultat d'expériences (10000 exécutions de l'algorithme de Knuth par graphique) faites sur des entiers de 1000 bits. Les graphiques représentent la densité de probabilité empirique du nombre de divisions euclidiennes effectuées par l'algorithme lors de plusieurs phases.

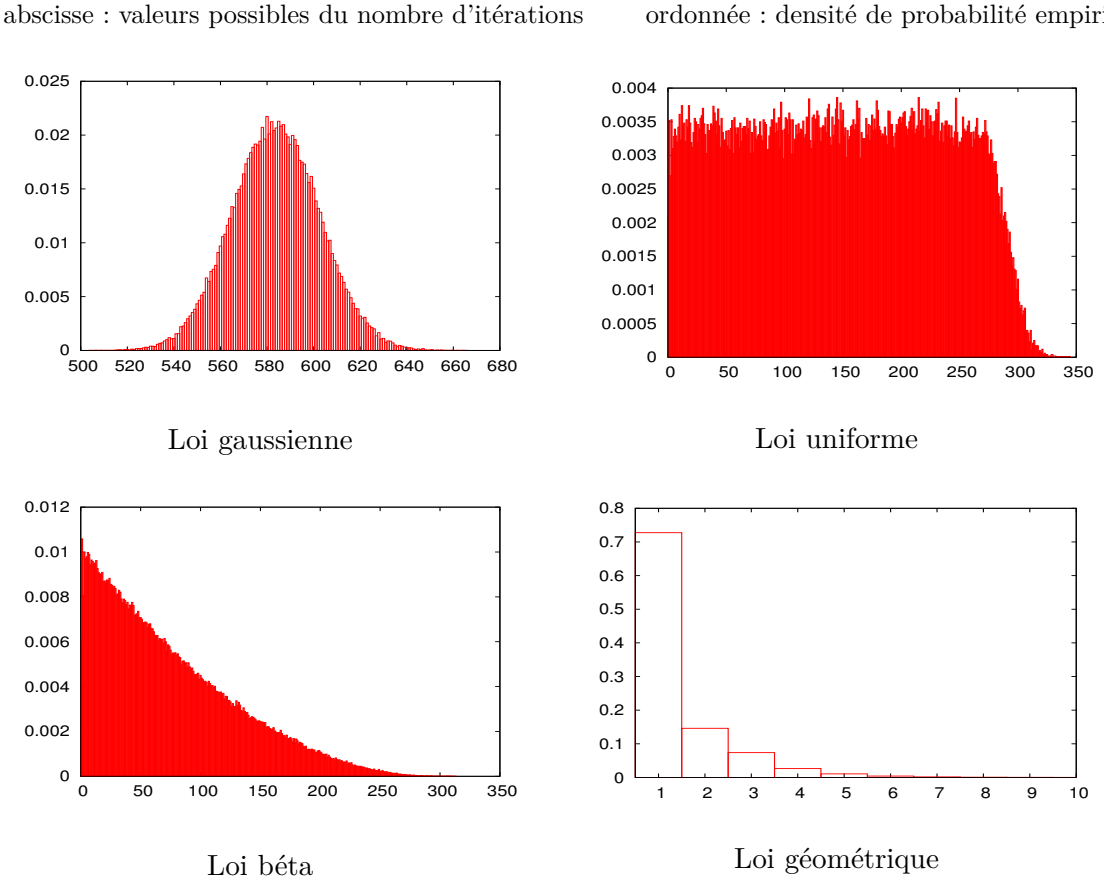


FIGURE 2.3 – Exemples de lois limites avec leur densité.

Les deux graphiques en haut de la figure 2.3 sont dédiés au cas $d = 1$ (2 entiers ou polynômes en entrée). L'algorithme de Knuth est alors exactement l'algorithme d'Euclide. Toutefois, nous considérons les deux notions de taille introduites à la section 2.3.2. A gauche, la taille de la paire (u, v) est donnée par la taille max (modèle associé à l'algorithme de Brun). C'est la notion de

taille usuelle dans les précédentes analyses de l'algorithme d'Euclide. A droite, la taille de (u, v) est la taille totale (modèle associé à l'algorithme de Knuth). A notre grande surprise, la loi limite est différente dans les deux modèles probabilistes. Avec la taille usuelle, nous obtenons une loi limite gaussienne comme il l'a été prouvé par Hensley dans [78]. Les lois limites gaussiennes sont courantes en analyses d'algorithmes, y compris pour les algorithmes du PGCD. La densité limite est alors définie sur $\mathbb{R}$ par $f(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$. Cependant dans le modèle de Knuth, la loi limite est une loi uniforme dont la densité est donnée par $f(x) = \mathbf{1}_{[0,1]}(x)$. C'est un cas particulier de la loi béta de paramètres $(1, 1)$. La densité d'une loi béta de paramètres (a, b) est définie sur $[0, 1]$ par

$$\beta_{a,b}(x) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} x^{a-1}(1-x)^{b-1}\mathbf{1}_{[0,1]}(x).$$

Les graphiques en bas de la figure 2.3 correspondent à l'algorithme de Knuth agissant sur 4 entrées ($d = 3$). A gauche, le graphique représente la densité empirique du nombre d'itérations pendant la première phase (paramètre L_1) alors que le graphique à droite correspond à la deuxième phase (paramètre L_2). Le théorème 7 indique que la densité à gauche est celle d'une loi béta de paramètre $(1, d)$ alors qu'à droite, nous obtenons une loi discrète quasi-géométrique. Un paramètre X suit une loi limite géométrique de paramètre $r \in [0, 1]$ si

$$\lim_{n\to\infty} \mathbb{P}_n [X > m] = r^m \qquad \text{pour } m \in \mathbb{N}.$$

La loi géométrique fait intervenir une puissance *exacte*. Nous obtenons aussi des lois limites où la puissance n'apparait qu'asymptotiquement lorsque m tend vers l'infini. Nous disons que X suit une loi limite quasi-géométrique de paramètre $r \in [0, 1]$ si pour $m \in \mathbb{N}$,

$$\lim_{n\to\infty} \mathbb{P}_n [X > m] = a_m \qquad \text{avec} \qquad a_m = b \cdot r^m \left(1 + O(\rho^{-m})\right),$$

b un réel positif et $\rho \in [0, 1[$. Autrement dit, a_m se comporte comme une quasi-puissance lorsque m grandit.

Les lois bétas sont omniprésentes en probabilité lors de l'étude des statistiques d'ordre. Étant données p variables i.i.d dans l'intervalle $[0, 1]$, la statistique d'ordre de rang k (k-ème plus petite valeur) suit une loi béta de paramètres $(k, p-k+1)$. Pour $k = 1$, il s'agit de la loi du minimum. Les urnes de Pólya [99] sont des modèles probabilistes basés sur le tirage aléatoire de boules avec replacement. Le nombre de boules d'une couleur donnée suit une loi béta-binomiale qui correctement normalisée et sous certaines conditions, converge vers une loi béta. Les lois bétas se retrouvent aussi lors de l'étude des arbres. Dans [28], Aldous décrit un modèle d'arbres aléatoires dont le partitionnement des sommets s'appuie sur la loi béta (beta-splitting model). Ce modèle est un cas particulier d'arbres aléatoires (random split trees) introduits par Devroye [46] qui utilisent des distributions générales et dont certains paramètres admettent des lois limites bétas. Toujours en lien avec les arbres, les auteurs de [100, 87] étudient les arbres croissants et montrent par exemple que la taille de certains sous-arbres suit asymptotiquement des lois bétas. Les exécutions de l'algorithme de tri Quicksort sont très liées au choix des pivots qui sont souvent des statistiques d'ordre. Les lois bétas ou des généralisations apparaissent alors naturellement lors des analyses ou comme lois limites [121, 77, 103, 38, 39, 101, 138, 139]. Les lois bétas se retrouvent dans l'analyse de paramètres qui ne sont pas liés à des statistiques d'ordre. Citons par exemples certaines grandeurs dans le processus d'orthogonalisation de Gram-Schmidt [26], l'étude des propriétés asymptotiques de certaines classes de graphes [34] ou l'étude de polynômes à racines unitaires [80]. Cette liste n'est bien entendu pas exhaustive.

2.3.3.4 Quid des analyses en distribution pour l'algorithme de Brun

Nous ne sommes pas parvenus à faire des analyses en distribution avec l'algorithme de Brun. Une analyse dynamique en moyenne nécessite uniquement (ou quasi-uniquement) des connaissances spectrales assez basiques sur l'opérateur de transfert et ces résultats ont par exemple été obtenus pour l'algorithme de Brun par Broise-Alamichel et Guivarc'h dans [35]. En revanche, une analyse en distribution s'appuie sur des inégalités à la Dolgopyat [48, 49] de l'opérateur de transfert. Baladi et Vallée ont montré que ces inégalités existent pour l'algorithme d'Euclide [29]. Comme l'algorithme de Knuth s'appuie sur l'algorithme d'Euclide, il était possible d'exploiter les résultats de Baladi-Vallée dans le cadre de l'algorithme de Knuth. Cependant, les mêmes inégalités n'existent pas encore pour l'algorithme de Brun. C'est pourquoi nous nous sommes limités à des analyses en moyenne.

2.4 Ébauche de preuve

Nous donnons ici une idée très globale des preuves des différents théorèmes dans le cas des entiers. Celles-ci s'appuient sur l'analyse dynamique.

2.4.1 Établir les systèmes dynamiques et les trajectoires

Le système dynamique sous-jacent à l'algorithme de Knuth est la transformation de Gauss qui est le système dynamique de l'algorithme d'Euclide. L'algorithme d'Euclide étant un cas particulier de l'algorithme de Brun, il suffit de décrire le système dynamique de Brun. La procédure pour retrouver le système dynamique est souvent la même et s'appuie sur la projection définie sur les entrées ordonnées $\mathcal{C}^B_{(d)}$ par

$$\pi(\mathbf{u}) = \frac{1}{u_0}\mathtt{End}\,\mathbf{u} = \left(\frac{u_1}{u_0}, \frac{u_2}{u_0}, \dots, \frac{u_d}{u_0}\right).$$

Bien entendu, cette définition comme les suivantes s'étendent à tout élément de l'union disjointe $\bigoplus_{i=1}^{d} \mathcal{C}^B_{(i)}$ qui est l'ensemble sur lequel opère l'algorithme de Brun pendant son exécution. Nous rappelons qu'une étape élémentaire de l'algorithme de Brun s'écrit

$$U_{(d)}(\mathbf{u}) = \mathtt{InsDis}\,(u_0 \bmod u_1, \mathtt{End}\,\mathbf{u})\,.$$

Si $\mathbf{v} = U_{(d)}(\mathbf{u})$, alors le système dynamique de Brun est la transformation $V_{(d)}$ qui transforme $\pi(\mathbf{u})$ en $\pi(\mathbf{v})$. Elle est définie sur le simplexe

$$\mathcal{J}_{(d)} = \{\mathbf{x} = (x_1, x_2, \dots, x_d) \mid 1 \geq x_1 \geq x_2 \geq \cdots \geq x_d \geq 0\}$$

et vérifie

$$V_{(d)}(\mathbf{x}) = \mathtt{InsDis}\left(\left\{\frac{1}{x_1}\right\}, \mathtt{End}\,\mathbf{x}\right), \qquad V_{(d)}(\mathbf{0}^d) = \mathbf{0}^d$$

avec $\{x\}$ la partie fractionnaire de x. Pour $d = 1$, nous retrouvons la transformation de Gauss définie sur $[0, 1]$ par $V_{(1)}(x) = \{1/x\}$ et $V_{(1)}(0) = 0$.

La fonction `InsDis` cache en fait trois transformations distinctes selon qu'il y a une insertion, un reste nul ou une égalité. Ces trois cas définissent trois familles de branches inverses respectivement notées $\mathcal{H}_{(d)}$ pour une insertion, $\mathcal{Z}_{(d)}$ pour un reste nul et $\mathcal{S}_{(d)}$ pour une égalité. Nous posons $\mathcal{F}_{(d)} = \mathcal{Z}_{(d)} \cup \mathcal{S}_{(d)}$ les branches finales qui interviennent à la fin de chaque phase.

Les branches inverses correspondant aux exécutions de l'algorithme de Brun sont alors données par l'ensemble $\mathcal{B}_{(d)}$ défini par

$$\mathcal{B}_{(d)} = \mathcal{P}_{(d)} \circ \mathcal{P}_{(d-1)} \circ \ldots \circ \mathcal{P}_{(1)} = \mathcal{P}_{(d)} \circ \mathcal{B}_{(d-1)} \tag{2.5}$$

où les ensembles de branches inverses $\mathcal{P}_{(d-\ell)}$ sont celles utilisées par l'algorithme de Brun sur $\mathcal{C}^B_{(d-\ell)}$ pendant la phase $d-\ell+1$,

$$\mathcal{P}_{(d-\ell)} = \mathcal{H}^\star_{(d-\ell)} \circ \mathcal{F}_{(d-\ell)}. \tag{2.6}$$

Ce résultat s'applique à l'algorithme d'Euclide lorsqu'il agit sur des données ordonnées. Mais lors de l'exécution de l'algorithme de Knuth, les entrées ne sont pas ordonnées et un échange peut être effectué pour réordonner. Il y a trois cas : (*i*) les deux éléments sont égaux ($u_0 = u_1$) et une seule division avec un quotient égal à 1 est effectuée. C'est une branche inverse $h_=$ qui est alors utilisée. (*ii*) Les éléments sont ordonnés ($u_0 > u_1$). Une trajectoire *classique* sur des données ordonnées est alors effectuée. (*iii*) Les éléments ne sont pas ordonnés ($u_0 < u_1$). Ils sont échangés et une exécution *classique* s'applique. L'ensemble des branches inverses $\widetilde{\mathcal{P}}_{(1)}$ construites par l'algorithme d'Euclide sur des entrées non-ordonnées est donné par

$$\widetilde{\mathcal{P}}_{(1)} = \{h_=\} \bigoplus \mathcal{P}_{(1)} \bigoplus \mathcal{P}_{(1)}$$

où $\mathcal{P}_{(1)} \bigoplus \mathcal{P}_{(1)}$ signifie une union disjointe entre $\mathcal{P}_{(1)}$ et une *copie formelle* de $\mathcal{P}_{(1)}$. L'algorithme de Knuth étant une succession de d appels à l'algorithme d'Euclide, l'ensemble des branches inverses construites par l'algorithme de Knuth sur l'ensemble $\mathcal{C}^K_{(d)}$ est $\widetilde{\mathcal{P}}_{(1)} \times \ldots \widetilde{\mathcal{P}}_{(1)}$ (d fois).

Nous venons de décrire les trajectoires admissibles des systèmes dynamiques sous-jacents aux algorithmes. À chaque entrée correspond une trajectoire (une suite admissible de branches inverses) et un PGCD, ce qui définit une bijection.

2.4.2 Séries génératrices d'intérêt et opérateurs

Les séries génératrices bivariées (ordinaires et de Dirichlet) s'écrivent pour un coût C défini sur $\mathcal{C}_{(d)}$,

$$C(z,w) := \sum_{\mathbf{u}\in\mathcal{C}_{(d)}} z^{\mathrm{d}(\mathbf{u})} w^{C(\mathbf{u})} \quad \text{et} \qquad C(s,w) := \sum_{\mathbf{u}\in\mathcal{C}_{(d)}} \frac{w^{C(\mathbf{u})}}{\mathrm{d}(\mathbf{u})^s}, \tag{2.7}$$

avec

$$\mathrm{d}(\mathbf{u}) = \begin{cases} u_0 & \text{pour Brun sur les entiers,} \\ u_0 u_1 \ldots u_d & \text{pour Knuth sur les entiers,} \\ |u_0| + |u_1| + \cdots + |u_d| & \text{pour Knuth sur les polynômes.} \end{cases}$$

La fonction $\mathrm{d}(\mathbf{u})$ a été définie de manière à manipuler facilement les séries génératrices. Nous posons $C(z) = C(z,1)$ et $C(s) = C(s,1)$ qui sont deux séries qui ne dépendent pas du paramètre étudié. La série génératrice du premier moment $C^{[1]}$ est la dérivée formelle par rapport à w en $w = 1$. Rappelons que les coefficients des séries génératrices sont en lien avec les grandeurs probabilistes du paramètre C à travers les relations (1.6), (1.7) et (1.9) dans le cas des polynômes. Dans le cas des entiers, il faut un peu adapter les formules et remplacer $[z^n]C$ par la somme cumulée $\Phi_n[C]$ (voir formule de Perron, Théorème 2).

Exprimer les séries génératrices avec les opérateurs de transfert est un moment clé de la partie formelle de l'analyse dynamique. Cette connexion s'appuie sur les ensembles de branches inverses décrits à la section précédente, sur les dictionnaires de l'analyse dynamique (voir section

1.3.5) et sur une propriété remarquable qui relie le jacobien au plus grand entier de l'entrée. Précisément si h est la branche inverse construite par l'algorithme de Brun sur l'entrée ordonnée $\mathbf{u} = (u_0, \dots, u_d) \in \mathcal{C}^B_{(d)}$ et si $g = \mathrm{PGCD}(\mathbf{u})$, alors $h(0) = \mathbf{u}/g$ et

$$\frac{1}{u_0^{d+1}} = \frac{\mathrm{Jac}[h](0)}{g^{d+1}} = \frac{1}{\mathrm{D}[h](0)^{d+1} g^{d+1}}.$$

En combinant tous ces éléments, nous obtenons une expression des séries génératrices en fonction des opérateurs de transfert donnée par la proposition suivante.

Proposition 1 (*a*) *Soit L_k^B le nombre d'itérations de l'algorithme de Brun pendant la phase k ($k \in [1, d]$) de l'algorithme. Alors la série génératrice bivariée $L_k^B(s, w)$ associée au paramètre L_k^B s'écrit sous la forme $L_k^B(s, w) = \zeta(s)\mathbf{B}_{(d),k,s,w}[\mathbf{1}](0)$ avec*

$$\mathbf{B}_{(d),k,s,w} = \left[\mathbf{P}_{(1),s} \circ \cdots \circ \mathbf{P}_{(d-k),s}\right] \circ \mathbf{P}_{(d-k+1),s,w} \circ \left[\mathbf{P}_{(d-k+2),s} \circ \cdots \circ \mathbf{P}_{(d),s}\right]. \tag{2.8}$$

Pour $\ell \in [0, d-1]$, les opérateurs $\mathbf{P}_{(d-\ell),s,w}$ s'écrivent

$$\mathbf{P}_{(d-\ell),s,w} = w\mathbf{F}_{(d-\ell),s} \circ (\mathbf{1} - w\mathbf{H}_{(d-\ell),s})^{-1}$$

avec $\mathbf{P}_{(d-\ell),s} = \mathbf{P}_{(d-\ell),s,1}$. Les opérateurs $\mathbf{F}_{(d-\ell),s}$ et $\mathbf{H}_{(d-\ell),s}$ sont les opérateurs de transfert (non pondérés) associés aux branches inverses $\mathcal{F}_{(d-\ell)}$ et $\mathcal{H}_{(d-\ell)}$ de la phase $\ell + 1$.

(*b*) *Soit L_k^K le nombre d'itérations de l'algorithme de Brun pendant la phase k ($k \in [1, d]$) de l'algorithme agissant sur les entiers. Alors la série génératrice bivariée $L_k^K(s, w)$ associée au paramètre L_k^K s'écrit sous la forme*

$$L_k^K(s, w) = \zeta(s)^{d+1} \cdot \frac{\zeta((k+1)s)}{\zeta(ks)\zeta(s)} \cdot \widetilde{\mathbf{P}}_{s,ks,w}[\mathbf{1}](0) \tag{2.9}$$

avec $\widetilde{\mathbf{P}}_{s,t,w} = \frac{1}{2} \cdot w \cdot (\mathbf{1} - w \cdot \mathbf{H}_{(1),s+t})^{-1} \circ (\mathbf{H}_{(1),s} + \mathbf{H}_{(1),t})$.

Dans tous les cas, les séries génératrices sont semblables. Elles se décomposent en deux parties : une qui est indépendante de w et une autre, qui fait intervenir l'opérateur de la phase k et qui dépend de w. Les opérateurs de la phase k dépendent tous d'un quasi-inverse de la forme $(\mathbf{1} - w\mathbf{G})^{-1}$ pour un opérateur $\mathbf{G}$. Or pour retrouver la série du premier moment $L_k^{[1]}$, il faut dériver par rapport à w les séries bivariées. La dérivée de $(\mathbf{1} - w\mathbf{G})^{-1}$ est donnée par $\mathbf{G} \circ (\mathbf{1} - w\mathbf{G})^{-2}$ qui fait apparaitre deux quasi-inverses. Nous pouvons écrire la proposition très informelle suivante.

Proposition 2 (*a*) *Dans tous les cas, la série génératrice $L_k^{[1]}(s)$ s'exprime en fonction des opérateurs de transfert. Cette expression fait intervenir deux quasi-inverses de la phase k, et un seul pour chaque autre phase.*

(*b*) *Dans tous les cas, la série génératrice $L_k(s, 1)$ s'exprime en fonction des opérateurs de transfert. Cette expression fait intervenir un quasi-inverse de la phase k, et un seul pour chaque autre phase.*

A ce stade de la méthode, les séries génératrices ont toutes une expression « explicite » en fonction d'opérateurs de transfert. Ces expressions ont été obtenues à l'aide des trajectoires ou branches inverses admissibles et des dictionnaires de l'analyse dynamique. Cela conclut la partie symbolique de la méthode.

2.4.3 Propriétés spectrales et analytiques des opérateurs et des séries génératrices

Les séries génératrices sont maintenant vues comme des objets analytiques. Dans le cas des entiers, ces dernières dépendent des opérateurs de transfert ce qui nous ramène à étudier les propriétés analytiques des opérateurs.

2.4.3.1 Propriétés spectrales et analytiques des opérateurs

Les opérateurs générateurs qui interviennent dans les expressions des séries génératrices sont principalement construits à partir de quelques opérateurs de base $\mathbf{F}_{(\ell),s}$, $\mathbf{H}_{(\ell),s}$, $\mathbf{H}_s = \mathbf{H}_{(1),s}$ et les quasi-inverses $(\mathbf{1} - \mathbf{H}_{(\ell),s})^{-1}$ pour $\ell \in [1, d]$.

Les opérateurs $\mathbf{F}_{(\ell),s}$ et $\mathbf{H}_{(\ell),s}$ sont tous analytiques sur le domaine $\Re s > 1$ et jouent un rôle secondaire [1] dans les propriétés analytiques des séries génératrices. De leurs côtés, les quasi-inverses ont un rôle essentiel. Si 1 est une valeur propre de $\mathbf{H}_{(\ell),s}$, alors $\mathbf{1} - \mathbf{H}_{(\ell),s}$ n'est pas injectif et le quasi-inverse $(\mathbf{1} - \mathbf{H}_{(\ell),s})^{-1}$ n'est pas défini. Nous sommes donc ramenés à étudier le spectre de $\mathbf{H}_{(\ell),s}$.

Dans [35], Broise-Alamichel et Guivarc'h étudient le spectre de ces opérateurs pour les algorithmes de Jacobi-Perron et Brun. Ils montrent que sur un espace fonctionnel adéquat, l'opérateur $\mathbf{H}_{(\ell),s}$ satisfait une propriété de Perron-Frobenius (voir la propriété 1) à savoir qu'il admet une unique valeur propre dominante $\lambda_\ell(s)$, simple et positive lorsque $s > 1$, isolée du reste du spectre par un saut spectral. Cette propriété spectrale entraine une décomposition des itérés de l'opérateur ainsi que du quasi-inverse sous la forme

$$\mathbf{H}^n_{(\ell),s} = \lambda_\ell(s)^n \mathbf{Q}_{(\ell),s} + \mathbf{R}^n_{(\ell),s} \quad (\mathbf{1} - \mathbf{H}_{(\ell),s})^{-1} = \frac{\lambda_\ell(s)}{1 - \lambda_\ell(s)} \mathbf{Q}_{(\ell),s} + (\mathbf{1} - \mathbf{R}_{(\ell),s})^{-1} \tag{2.10}$$

avec $\mathbf{Q}_{(\ell),s} \circ \mathbf{R}_{(\ell),s} = \mathbf{R}_{(\ell),s} \circ \mathbf{Q}_{(\ell),s} = 0$, $||\mathbf{Q}^n_{(\ell),s}|| = O(\rho_\ell^n |\lambda_\ell(s)|^n)$ et $\rho_\ell < 1$ (uniforme en s dans chaque compact). Or en $s = \ell + 1$, l'opérateur $\mathbf{H}_{(\ell),\ell+1}$ est un transformateur de densité et sa valeur propre dominante vaut 1. En utilisant des propriétés de décroissance du rayon spectral de $\mathbf{H}_{(\ell),s}$ à la fois sur l'axe réel et sur les droites verticales $\{\Re s = \sigma\}$ ($\sigma > 1$), nous obtenons que le quasi-inverse $(\mathbf{1} - \mathbf{H}_{(\ell),s})^{-1}$ est analytique sur le demi-plan $\{\Re s \geq \ell + 1\}$ excepté en $s = \ell + 1$ où il admet un pôle simple. Nous pouvons maintenant transférer cette propriété aux séries génératrices et nous obtenons la proposition informelle suivante.

Proposition 3 (*a*) *Dans le cas de l'algorithme de Brun, les séries génératrices $L_k(s, 1)$ et $L_k^{[1]}(s)$ admettent un pôle dominant en $s = d + 1$. Ce pôle est simple pour $L_k(s, 1)$ quel que soit k. Il est aussi simple si $k \geq 2$ pour $L_k^{[1]}(s)$. En revanche, c'est un pôle double si $k = 1$.*

(*b*) *Dans le cas de l'algorithme de Knuth, les séries génératrices $L_k(s, 1)$ et $L_k^{[1]}(s)$ admettent un pôle dominant en $s = 1$. Ce pôle est d'ordre $d + 1$ pour $L_k(s, 1)$ quel que soit k. Il est aussi d'ordre $d + 1$ si $k \geq 2$ pour $L_k^{[1]}(s)$. En revanche, c'est un pôle d'ordre $d + 2$ si $k = 1$.*

2.4.4 Théorèmes de transfert et fin de l'analyse en moyenne

Les propriétés analytiques des séries génératrices étant connues, nous pouvons maintenant extraire les coefficients à l'aide des extracteurs. Quel que soit l'algorithme (Knuth ou Brun) ou le type d'entrées (entiers ou polynômes), nous sommes dans l'une des deux situations suivantes selon la phase étudiée :

1. la plupart du temps...

Cas $k \geq 2$: La position et l'ordre des pôles dominants des séries $L_k^{[1]}(z)$ et $L_k^{[1]}(s)$ sont identiques à la position et l'ordre des pôles dominants de $L_k(1,z)$ et $L_z(1,s)$.

Cas $k = 1$: La position des pôles dominants des séries $L_1^{[1]}(z)$ et $L_1^{[1]}(s)$ est identique à celle des pôles dominants de $L_k(1,z)$ et $L_z(1,s)$. En revanche, l'ordre des pôles dominants diffère de 1.

Les séries génératrices satisfont aussi les hypothèses du théorème de transfert donné au chapitre précédent (théorème 1) ou du théorème taubérien de Delange (théorème 3). En appliquant ces théorèmes aux séries génératrices, nous obtenons deux types d'asymptotiques :

Cas $k \geq 2$: Les coefficients des séries $L_k^{[1]}(z)$ et $L_k^{[1]}(s)$ ont le même comportement asymptotique (à une constante près) que ceux de $L_k(1,z)$ et $L_z(1,s)$.

Cas $k = 1$: Les coefficients des séries $L_1^{[1]}(z)$ et $L_1^{[1]}(s)$ ont un comportement asymptotique différent des coefficients de $L_k(1,z)$ et $L_z(1,s)$. Un terme linéaire en la taille n vient s'ajouter pour l'asymptotique des coefficients de $L_k^{[1]}(z)$ et $L_k^{[1]}(s)$.

Pour conclure, l'espérance de L_k se déduit des formules (1.6), (1.7) ou (1.9) qu'il faut adapter avec les séries de Dirichlet en remplaçant $[z^n]C(z)$ par $\Phi_n[C(s)]$ avec $C(s) = L_k(s,1)$ ou $C(s) = L_k^{[1]}(s)$. Le rapport entre les asymptotiques des coefficients de $L_k^{[1]}(s)$ et $L_k(s,1)$ conduit aux deux types de résultats :

Cas $k \geq 2$: L'espérance $\mathbb{E}_n[L_k]$ de L_k est d'ordre constant.

Cas $k = 1$: L'espérance $\mathbb{E}_n[L_1]$ de L_1 est d'ordre linéaire en la taille.

Bien entendu, il faut faire très attention aux constantes mais il s'agit essentiellement d'une difficulté calculatoire.

2.4.5 Vers une analyse en distribution

Lors de l'analyse en distribution des paramètres L_k pour l'algorithme de Knuth, nous nous sommes retrouvés dans une situation inédite en analyse dynamique. La série génératrice bivariée $L_k(s,w)$ est de la forme

$$L_k(s,w) = \zeta(s)^{d+1} \cdot w \cdot \frac{A(s,w)}{A(s,1)}$$

avec

$$A(s,w) = \frac{1}{w}\widetilde{\mathbf{P}}_{s,ks,w}[\mathbf{1}](0) \quad \text{et} \quad A_m(s) = \mathbf{H}^m_{(k+1)s}[\varphi_{s,ks}](0).$$

Pour $k = 1$, $A(s,1)$ admet un pôle simple en $s = 1$ alors que pour $k \geq 2$, $A(s,1)$ est analytique en $s = 1$. Les fonctions zêta admettent un pôle simple en $s = 1$.

Lorsque w varie, la singularité dominante de $L_1(s)$ change de nature. En $w = 1$, $s = 1$ est un pôle d'ordre $d + 1$. Pour $w > 1$, $A(s,w)$ admet un pôle en $s = \sigma(w) > 1$ qui est un pôle dominant simple de $L_1(s,w)$. Pour $w < 1$, $A(s,w)$ admet un pôle en $s = \sigma(w) < 1$ mais le pôle dominant de $L_1(s)$ est 1 qui est d'ordre d. Il y a donc des transitions de phase dans la nature de la singularité lorsque $k = 1$. Ces transitions de phase n'existent plus lorsque $k \geq 2$. On retrouve cette notion de transitions de phase dans le livre de Flajolet-Sedgewick [60] (page 704).

Dans [1], nous avons décrit deux nouveaux extracteurs (un pour chaque type de série génératrice) adaptés à ce contexte et qui conduisent à des lois limites bétas, géométriques ou quasi-géométriques. Nous ne détaillons pas ces résultats ici mais ils sont une sorte d'équivalents du théorème des quasi-puissances de Hwang [79] qui conduit à des lois limites gaussiennes.

2.4.6 Conclusion

Nous en avons terminé avec la preuve des résultats pour les seuls paramètres L_k. Nous avons décrit assez précisément toutes les étapes de l'analyse dynamique en omettant volontairement de nombreux détails des preuves. La prochaine section résume d'autres résultats obtenus sur d'autres paramètres des algorithmes d'Euclide ou de Knuth ou bien sur un algorithme de type diviser pour régner.

2.5 Autres résultats

2.5.1 Algorithme d'Euclide et de Knuth

Lors des analyses des algorithmes d'Euclide et de Knuth, nous avons mis en évidence d'autres lois limites.

Algorithme d'Euclide : dans [18, 6], nous avons décrit les lois limites de la taille des continuants à une fraction de l'exécution et de la complexité binaire. Nous reprenons les notations de la section 2.3.1.1. Étant donné un rationnel r, la taille du continuant à une fraction r de l'exécution est donnée par la taille binaire ou le degré de $a_i(\mathbf{u})$ avec $i = \lfloor r \cdot L(\mathbf{u}) \rfloor$. Nous avons démontré que la taille des continuants décroit très régulièrement pendant l'exécution de l'algorithme. En moyenne, la taille est donnée par $(1-r)\cdot n$ avec n la taille-max initiale. La variance est aussi linéaire et nous avons même démontré une loi limite gaussienne.

La complexité binaire naïve d'une division euclidienne $a = bq+r$ est d'ordre $O(\ell(q)\ell(b))$ avec $\ell(x)$ la taille de x (taille binaire ou degré). En sommant sur toutes les étapes $\ell(b_i(\mathbf{u}))\ell(q_i(\mathbf{u}))$, nous obtenons la complexité binaire de l'algorithme d'Euclide. Nous avons montré que la complexité binaire de l'algorithme d'Euclide sur les polynômes suit asymptotiquement une loi limite gaussienne d'espérance quadratique en la taille de l'entrée et de variance cubique. Nous avons obtenu le même résultat avec l'algorithme d'Euclide étendu qui calcule en plus les coefficients de Bezout. La différence est que la loi limite existe aussi pour l'algorithme étendu sur les entiers.

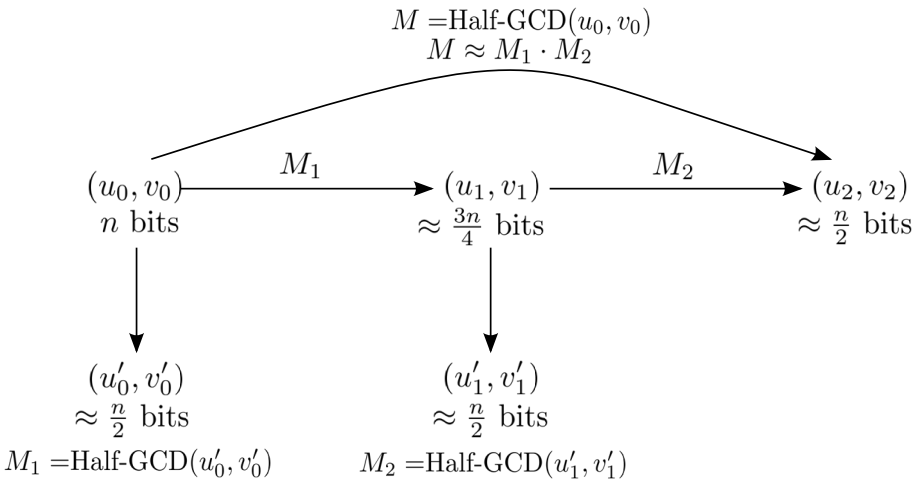
Algorithme de Knuth : afin d'étudier l'évolution des données, nous nous sommes intéressés à la taille des PGCD intermédiaires lors de l'exécution. Nous posons $D_k(\mathbf{u})$ la taille du PGCD de $(u_0, u_1, \ldots, u_k)$ pour $k = 0..d$. Alors les paramètres D_k suivent exactement les mêmes lois que les paramètres L_k, aux constantes près. En particulier, D_0 suit une loi limite béta de paramètre $(1, d)$ alors que D_k pour $k \geq 1$ suit une loi limite géométrique (cas des polynômes) ou quasi-géométrique (cas des entiers). L'espérance de D_0 est linéaire en la taille alors que les espérances des D_k sont asymptotiquement constantes.

2.5.2 Algorithme de Knuth-Schönhage

L'analyse dynamique est bien adaptée pour l'étude probabiliste des algorithmes du PGCD basés uniquement sur des divisions (division euclidienne, paire, impaire, par excès, par défaut, binaire, etc.). Comme la taille cumulée des restes successifs est d'ordre quadratique, ces algorithmes sont tous au moins quadratiques. Afin d'accélérer les divisions, Lehmer a eu l'idée de n'utiliser que les bits dominants des entiers afin de déterminer rapidement les quotients [93]. En utilisant récursivement cette idée, Knuth [84] puis Schönhage [119] en 1971 ont créé un algorithme de type diviser pour régner qui utilise les multiplications rapides pour obtenir des algorithmes sous-quadratiques.

L'algorithme de Knuth-Schönhage s'appuie sur une fonction récursive, appelée `Half-GCD`(u, v), qui pour des entiers (u, v) de n bits calcule des restes intermédiaires (u'', v'') dans l'exécution

de l'algorithme d'Euclide et ayant environ $n/2$ bits. `Half-GCD` calcule aussi une matrice M permettant de passer de (u, v) à (u'', v''). Pour cela, la fonction ne conserve que la moitié des bits de (u, v) et obtient un premier couple d'entiers (u_1, v_1) ayant $n/2$ bits. Un premier appel récursif avec (u_1, v_1) calcule une première matrice M_1. Le critère de Jebelean dit alors que la matrice M_1, appliquée aux grands entiers (u, v), conduit à un couple d'entiers (u', v') qui sont des restes intermédiaires de taille environ $3n/4$ bits. C'est à ce stade que des multiplications rapides sont utilisées. La méthode est appliquée une deuxième fois avec (u', v'). Tout d'abord, les $n/2$ premiers bits de (u', v') sont conservés et définissent un couple d'entiers (u_2, v_2). La fonction `Half-GCD` appliquée à (u_2, v_2) retourne une matrice M_2 et le critère de Jebelean s'applique encore. En multipliant M_2 (avec des multiplications rapides) et (u', v'), nous obtenons le couple (u'', v'') de restes intermédiaires ayant environ $n/2$ bits.



Dans l'article court [10] et sa version longue [2], nous avons analysé en moyenne la complexité binaire de l'algorithme de Knuth-Schönhage. Nous montrons que la complexité moyenne sur des entiers de taille n est d'ordre $O(\mu(n) \log n)$ avec $\mu(n)$ la complexité binaire de la multiplication utilisée sur des entiers de taille n. Ce résultat est valable même pour des multiplications quasi-linéaires de type FFT.

Les analyses s'appuient sur une régularité forte des données à une fraction de l'exécution comme cela a été expliqué à la section 2.5.1. Précisément, chaque appel récursif est vu comme une partie de l'exécution de l'algorithme d'Euclide classique : les quotients sont les mêmes mais les entiers utilisés sont plus petits (car chaque appel récursif ne conserve que les bits dominants). Il est donc possible de lier la complexité d'un appel récursif à la complexité d'une partie de l'algorithme d'Euclide. Nous avons décrit très précisément l'évolution de la distribution des données au cours de l'algorithme d'Euclide. Le résultat obtenu est plus précis que celui de la section 2.5.1 car les fractions dépendent de la taille des entrées. Nous avons mis en évidence une très forte régularité des appels récursifs ce qui a conduit pour la première fois à une analyse précise d'un algorithme de type diviser pour régner.

2.6 Conclusion et perspectives de recherche

Dans ce chapitre, nous avons présenté les analyses de deux algorithmes du PGCD agissant sur plusieurs entrées entières ou polynomiales. Nous avons réalisé une analyse en distribution complète du nombre d'itérations de l'algorithme de Knuth et nous nous sommes limités à une

analyse en moyenne de l'algorithme de Brun. Pour $d = 1$, les algorithmes de Knuth et de Brun correspondent à l'algorithme d'Euclide et tous les résultats trouvés s'appliquent. Ces travaux constituent les premières analyses probabilistes d'algorithmes du PGCD en dimensions supérieures et s'appuient sur les connaissances acquises (et l'expérience) en petites dimensions.

Notre objectif est de réaliser une étude systématique et précise des algorithmes du PGCD. Les travaux sur les algorithmes de Knuth et Brun peuvent être étendus selon plusieurs axes : l'étude d'autres algorithmes, l'étude d'autres paramètres et adapter l'analyse dynamique en distribution aux algorithmes en dimensions supérieures.

D'autres stratégies que celles utilisées par les algorithmes de Knuth et de Brun ont été conçues, en particulier dans le cadre des approximations diophantiennes simultanées. On peut citer par exemple les algorithmes de Jacobi-Perron ou de Selmer-Euclide dont les systèmes dynamiques ont été étudiés. Il serait intéressant dans un premier temps de réaliser des analyses en moyenne de ces algorithmes afin de continuer les comparaisons. Une des difficultés dans ce travail sera de savoir si les résultats sur les systèmes dynamiques sous-jacents sont suffisants pour réaliser une analyse en moyenne ou bien s'il faudra adapter les approches existantes.

A plus long terme, il serait intéressant de réaliser des analyses en distribution de ces algorithmes. Si nous adoptons la même approche que pour deux entrées, cela suppose d'étendre les travaux de Baladi-Vallée [29] en obtenant des bornes à la Dolgopyat sur les opérateurs de transfert. Ce type de résultats présente un double intérêt à la fois pour l'analyse d'algorithmes mais aussi pour la théorie des systèmes dynamiques.

Un autre axe de travail est l'étude d'autres paramètres. L'analyse en moyenne de la complexité binaire des algorithmes de Brun et de Knuth est probablement accessible très facilement compte tenu des travaux réalisés jusqu'à maintenant. L'algorithme de Knuth est finalement assez efficace pour calculer le PGCD mais il n'a pas été conçu pour calculer des coefficients de Bezout de petites tailles. A l'inverse, les algorithmes issus des approximations diophantiennes simultanées sont probablement plus intéressants pour calculer des petits coefficients de Bezout mais l'analyse de l'algorithme de Brun semblent montrer qu'ils sont moins efficaces pour le calcul de PGCD. Ces remarques se basent sur des expériences ou des intuitions mais aucune analyse probabiliste ne vient les confirmer (ou les infirmer !). Un axe de travail serait d'analyser des paramètres tels que la taille des coefficients de Bezout ou la qualité d'approximation simultanée afin d'avoir un autre point de vue sur ces algorithmes.

Chapitre 3

Réseaux euclidiens et modélisations de l'algorithme LLL

Un réseau euclidien est un ensemble de points régulièrement répartis dans l'espace. Les algorithmes de réduction de réseaux visent à trouver une représentation matricielle des réseaux avec des vecteurs assez courts en norme à partir d'une représentation matricielle quelconque.

Ce chapitre résume des travaux commencés pendant la thèse de Mariya Georgieva [64] que nous continuons d'affiner aujourd'hui. J'ai co-encadré la thèse de Mariya avec Julien Clément et Brigitte Vallée. Actuellement, nous co-encadrons avec Julien la thèse de Dimitri Darthenay dont le début portait sur une généralisation des résultats de Mariya à la dimension supérieure. Un article est en cours de rédaction sur les travaux de ce chapitre.

Les algorithmes du PGCD peuvent être vus comme des algorithmes de réduction de réseaux agissant sur plusieurs vecteurs entiers de dimension 1 (autrement dit des entiers). Les divisions euclidiennes correspondent à des translations qui visent à réduire la norme des vecteurs. Les algorithmes de réduction de réseaux agissent sur des vecteurs de dimensions supérieures et utilisent aussi des translations pour obtenir des vecteurs courts. Ce chapitre s'inscrit ainsi dans la continuité des travaux sur les algorithmes du PGCD mais avec un degré de généralisation supplémentaire.

L'objectif à long terme serait de réaliser une analyse en moyenne systématique des algorithmes de réduction de réseaux. Nous nous intéressons ici uniquement à l'algorithme LLL [95], introduit en 1982 par A.K. Lenstra, H.W. Lenstra et L. Lovász, dans le cadre de la factorisation de polynômes à coefficients rationnels. C'est le premier algorithme de réduction de réseaux de complexité polynomiale et il s'applique maintenant à de nombreux domaines comme la programmation entière, la cryptographie, l'arithmétique ou encore la théorie de la complexité (voir le livre [104] dédié à LLL et les références à l'intérieur). Malheureusement, la dynamique de l'algorithme LLL est trop complexe et les techniques d'analyses d'algorithmes ne s'appliquent plus lorsque la dimension du réseau est plus grande que 3. Nous adoptons dans ce chapitre une approche légèrement différente de celle utilisée pour les algorithmes du PGCD. Nous commençons par modéliser l'algorithme en simplifiant ses exécutions. Nous espérons ainsi que les modèles simplifiés conservent certaines propriétés de l'algorithme initial et nous analysons les modèles simplifiés.

Plan. La section 3.1 introduit les notions élémentaires autour des réseaux euclidiens comme la base, le potentiel ou encore le principe de réduction d'un réseau. L'algorithme LLL y est décrit ainsi que les analyses probabilistes connues. La section 3.2 est dédiée à la présentation des modèles d'exécution de LLL mais aussi à un modèle aléatoire de réseaux euclidiens suffisamment

général pour capturer de nombreuses applications. Nous terminons avec les résultats obtenus autour de chaque modèle d'exécution (section 3.3).

3.1 Réseaux euclidiens et algorithme LLL

3.1.1 Réseaux euclidiens, bases et réduction

Un réseau euclidien $\mathcal{L}$ de $\mathbb{R}^n$ est un sous-groupe discret additif de $\mathbb{R}^n$. Il existe une famille libre $\mathbf{b} = (b_1, \ldots, b_d)$ de vecteurs de $\mathbb{R}^n$ telle que tout vecteur de $\mathcal{L}$ s'écrit comme une combinaison linéaire entière des vecteurs de $\mathbf{b}$, soit

$$\mathcal{L} := \mathcal{L}(\mathbf{b}) = \mathbb{Z} \cdot b_1 + \mathbb{Z} \cdot b_2 + \cdots + \mathbb{Z} \cdot b_d.$$

Un tel système de vecteurs est appelé base du réseau. Un réseau admet une infinité de bases et elles ont toutes le même cardinal d, appelé *dimension* du réseau.

Toutes les bases n'ont pas les mêmes propriétés euclidiennes. Les bases avec des vecteurs *assez courts* et *assez orthogonaux* sont bien adaptées pour avoir une algorithmique efficace sur les réseaux. La réduction d'un réseau $\mathcal{L}$ consiste à calculer une base $\mathbf{b}'$ avec des vecteurs *assez courts* et *assez orthogonaux* à partir d'une base $\mathbf{b}$ quelconque de $\mathcal{L}$. La base $\mathbf{b}'$ est appelée base réduite de $\mathcal{L}$. En petites dimensions [122, 130], il existe une définition « naturelle » de base réduite qu'il est possible de construire efficacement. En dimensions supérieures, les notions de bases réduites sont multiples (bases HKZ-réduites [86, 23, 90], BKZ-réduites [118],...) et correspondent à des propriétés euclidiennes différentes.

Réduire fortement un réseau est une tâche difficile. Le problème SVP (Shortest Vector Problem) qui consiste à calculer un vecteur non-nul le plus court du réseau est un problème conjecturé NP-difficile mais démontré NP-difficile avec des réductions randomisées [22]. Autrement dit, calculer une base qui contient un des vecteurs les plus courts est déjà un problème difficile.

Dans ce chapitre, nous étudions l'algorithme LLL qui produit des bases dites LLL-réduites. Une base LLL-réduite résout le problème SVP mais à un facteur d'approximation exponentiel près. Nous introduisons maintenant l'orthogonalisation de Gram-Schmidt sur laquelle s'appuie LLL.

3.1.2 Orthogonalisation de Gram-Schmidt

Etant donnée une base $\mathbf{b}$ du réseau $\mathcal{L}(\mathbf{b})$, l'algorithme LLL commence par calculer la base orthogonale de Gram-Schmidt $\mathbf{b}^\star = (b_1^\star, \ldots, b_d^\star)$ de $\mathbf{b}$ et la matrice de passage M telle que $\mathbf{b} = M\mathbf{b}^\star$. Par définition, les vecteurs de $\mathbf{b}$ sont orthogonaux et pour $i = 1..d$, l'espace vectoriel engendré par $(b_1, \ldots, b_i)$ est identique à celui engendré par les vecteurs orthogonaux $(b_1^\star, \ldots, b_i^\star)$. La matrice $M = (m_{i,j})_{i,j=1..d}$ est donnée par

$$m_{i,i} = 1, \quad m_{i,j} = 0 \ \text{ for } j > i, \qquad m_{i,j} = \frac{(b_i \cdot b_j^\star)}{||b_j^\star||^2} \ \text{ for } i > j$$

avec $(u \cdot v)$ le produit scalaire entre les vecteurs u et v. C'est une matrice triangulaire inférieure avec des 1 sur la diagonale (voir figure 3.1). Néanmoins, la base $\mathbf{b}^\star$ n'est en général pas une base du réseau $\mathcal{L}(\mathbf{b})$.

Les coefficients de la matrice ont un rôle essentiel dans l'algorithme LLL. Afin de contrôler leur taille, il est possible de travailler avec des valeurs bornées. Nous dirons que le coefficient

$$M := \begin{array}{c} \\ b_1 \\ b_2 \\ \vdots \\ b_{i-1} \\ b_i \\ b_{i+1} \\ \vdots \\ b_d \end{array} \begin{array}{c} \begin{array}{cccccccc} b_1^\star & b_2^\star & \dots & b_{i-1}^\star & b_i^\star & b_{i+1}^\star & \dots & b_d^\star \end{array} \\ \begin{pmatrix} 1 & 0 & \dots & 0 & 0 & 0 & 0 & 0 \\ m_{2,1} & 1 & \dots & 0 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ m_{i-1,1} & m_{i-1,2} & \dots & 1 & 0 & 0 & 0 & 0 \\ m_{i,1} & m_{i,2} & \dots & m_{i,i-1} & 1 & 0 & 0 & 0 \\ m_{i+1,1} & m_{i+1,2} & \dots & m_{i+1,i-1} & m_{i+1,i} & 1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ m_{d,1} & m_{d,2} & \dots & m_{d,i-1} & m_{d,i} & m_{d,i+1} & \dots & 1 \end{pmatrix} \end{array}$$

FIGURE 3.1 – Matrice de passage M pour l'orthogonalisation de Gram-Schmidt.

$m_{i,j}$ est propre si $|m_{i,j}| < \frac{1}{2}$. Le vecteur b_i est propre si tous les coefficients $(m_{i,j})_{j=1..i-1}$ le sont. Finalement, la base $\mathbf{b}$ et la matrice M seront dites propres si tous les vecteurs $(b_1, \dots, b_d)$ le sont. Il est possible de *proprifier* (rendre propre) un coefficient, une ligne ou la matrice M avec uniquement des translations. Par exemple, l'opération

$$b_i \leftarrow b_i - \lfloor m_{i,j} \rceil b_j$$

avec $\lfloor x \rceil$ le plus proche entier de x, transforme le coefficient $m_{i,j}$ en sa partie fractionnaire centrée $\{\{m_{i,j}\}\} = m_{i,j} - \lfloor m_{i,j} \rceil$ qui est par définition dans l'intervalle $[-\frac{1}{2}, \frac{1}{2}]$. Il suffit d'itérer cette opération pour proprifier une ligne ou la matrice complète. Un point important qui servira dans nos modélisations est que les translations ne modifient pas la base orthogonale $\mathbf{b}^\star$ ni la norme des orthogonalisés $(||b_1^\star||, \dots, ||b_d^\star||)$.

Pour une base $\mathbf{b}$ du réseau $\mathcal{L}(\mathbf{b})$, le déterminant de $\mathcal{L}(\mathbf{b})$ est défini par

$$\det \mathcal{L}(\mathbf{b}) = \prod_{i=1}^{d} ||b_i^\star||.$$

Le déterminant est un invariant du réseau c'est-à-dire qu'il ne dépend pas de la base $\mathbf{b}$. Nous utilisons le déterminant pour définir correctement le potentiel des modèles d'exécutions de LLL.

3.1.3 Principes de l'algorithme LLL

L'algorithme LLL dépend d'un paramètre $\tau \leq 1$ et construit une base dite τ-LLL-réduite du réseau. Une base $\mathbf{b}$ est τ-LLL-réduite si elle est propre et si les conditions de Lovász $\mathcal{C}_\tau(i)$ pour $i = 1..d-1$ sont satisfaites. La condition de Lovász $\mathcal{C}_\tau(i)$ est définie par

$$\mathcal{C}_\tau(i) : \qquad x_i + m_{i+1,i}^2 \geq \tau^2 \qquad \text{où} \qquad x_i := \frac{||b_{i+1}^\star||^2}{||b_i^\star||^2}. \tag{3.1}$$

Cette condition assure qu'à la sortie de l'algorithme, les rapports x_i ne sont pas trop petits. L'algorithme LLL ne fait que deux types d'opérations : des translations pour rendre propre la base et des échanges entre deux vecteurs consécutifs.

(*i*) *Les translations* : la translation $b_i - \lfloor m_{i,j} \rceil b_j$ rend propre le coefficient $m_{i,j}$ de la matrice de Gram-Schmidt. Ces translations assurent que la base sera propre à la fin du processus et

permettent de contrôler la taille de la matrice de Gram-Schmidt pendant l'exécution. Les translations réduisent aussi la norme des vecteurs afin d'obtenir des vecteurs *assez courts*. En revanche, elles ne modifient pas la base orthogonale $\mathbf{b}^\star$ ni la norme des orthogonalisés $(||b_1^\star||, \ldots, ||b_d^\star||)$.

(*ii*) *Les échanges :* si la condition de Lovász $\mathcal{C}_\tau(i)$ n'est pas vérifiée entre les vecteurs (propres) b_i et b_{i+1}, alors l'algorithme échange b_i et b_{i+1}. Comme l'orthogonalisation de Gram-Schmidt dépend de l'ordre des vecteurs, l'échange modifie la base orthogonale $\mathbf{b}^\star$ et la norme des orthogonalisés $(||b_1^\star||, \ldots, ||b_d^\star||)$ mais il ne modifie pas les vecteurs de $\mathbf{b}$ (à l'ordre près). Nous notons $\check{\mathbf{b}}$ et $\check{\mathbf{b}}^\star$ les bases obtenues après l'échange. Des calculs techniques mais simples montrent que les normes des orthogonalisés et les rapports $\check{x}_i$ vérifient

$$||\check{\mathbf{b}}_i^\star||^2 = \rho||\mathbf{b}_i^\star||^2, \qquad ||\check{\mathbf{b}}_{i+1}^\star||^2 = \frac{1}{\rho}||\mathbf{b}_{i+1}^\star||^2, \qquad \check{m}_{i+1,i} = \frac{1}{\rho}m_{i+1,i}, \tag{3.2}$$

$$\check{x}_{i-1} = \rho x_{i-1}, \quad \check{x}_i = x_{i-1}/\rho^2, \quad \check{x}_{i+1} = \rho x_{i+1} \tag{3.3}$$

$$\text{avec} \qquad \rho = x_i + m_{i+1,i}^2 < \tau^2 \leq 1. \tag{3.4}$$

Les équations (3.2), (3.3) et (3.4) sont fondamentales pour montrer que l'algorithme se termine ainsi que pour sa modélisation. Le facteur ρ est appelé facteur de décroissance de l'algorithme. Si l'on considère le potentiel

$$Q(\mathbf{b}) := \prod_{i=1}^{d-1} ||\mathbf{b}_i^\star||^{2(d-i)},$$

alors les translations ne modifient pas ce potentiel (car elles ne modifient pas la base orthogonale). En revanche, chaque échange fait décroitre le potentiel d'un facteur $\rho < \tau^2 \leq 1$. La décroissance stricte du potentiel assure la terminaison de l'algorithme (avec des réseaux à coefficients entiers ou rationnels) et le nombre d'échanges $K_\tau(\mathbf{b})$ de l'algorithme sur le réseau $\mathcal{L}(\mathbf{b})$ est borné par

$$K_\tau(\mathbf{b}) \leq \frac{1}{2|\ln \tau|} \ln Q(\mathbf{b})$$

dès que $\mathbf{b}$ est à coefficients entiers et $\tau < 1$. Notons que $\ln Q(\mathbf{b})$ est quadratique en la dimension d. Si $\tau = 1$, il y a toujours une décroissance stricte mais les bornes connues sur le nombre d'itérations sont exponentielles en la dimension d [25]. LLL est succinctement décrit par l'algorithme 4.

Algorithme 4 : Algorithme LLL avec $\tau \leq 1$.

Entrée: Une base $\mathbf{b} = (b_1, \ldots, b_d)$ d'un réseau $\mathcal{L}$.
Sortie: Une base τ-LLL-reduite de $\mathcal{L}$.

```
Calculer la base orthogonale de Gram-Schmidt b* et la matrice M ;
Proprifier la base b ;
tant que b n'est pas τ-LLL-réduite faire
    Choisir i tel que C_τ(i) n'est pas vérifiée ;
    Échanger b_i et b_{i+1} ;
    Mettre à jour b* et M ;
    Proprifier b ;
fin tant que
retourner b
```

3.1.4 Analyses probabilistes connues de LLL

Il existe plusieurs analyses probabilistes de l'algorithme LLL qui sont plus ou moins précises selon la dimension d. Pour $d = 1$, l'algorithme LLL est en fait l'algorithme d'Euclide centré qui utilise la division centrée au lieu de la division euclidienne. Pour tout couple d'entiers (a, b) avec

$b \neq 0$, la division centrée calcule un couple (q, r) avec $a = b \cdot q + r$ et $-b/2 < r \leq b/2$. Le PGCD de a et b reste égal au PGCD de b et r.

L'analyse en moyenne du nombre d'itérations de l'algorithme centré a été réalisée par Rieger [112] en 1978. En utilisant les techniques d'analyse dynamique, Vallée et Akhavi ont réalisé l'analyse probabiliste des paramètres additifs (voir chapitre 2 section 2.3.1.1) et de la complexité binaire [27, 136]. Le système dynamique de l'algorithme centré entre aussi dans le cadre des travaux de Baladi-Vallée [29] qui ont montré que tous les coûts additifs à croissance modérée (dont le nombre d'itérations fait partie) satisfont une loi limite gaussienne.

Pour $d = 2$ et $\tau = 1$, l'algorithme LLL est exactement l'algorithme de Gauss qui semble avoir été décrit en premier par Lagrange. Lagarias [91] a donné la première borne polynomiale sur le nombre d'itérations de l'algorithme de Gauss dans le pire des cas. Cette borne a ensuite été améliorée par Vallée [131]. Dans [42], Daudet, Flajolet et Vallée ont réalisé une analyse en moyenne du nombre d'itérations avec le modèle aléatoire de la boule uniforme (2 vecteurs choisis uniformément dans la boule unité). L'inconvénient avec ce modèle est que les bases en entrée sont presque réduites. Dans [135, 134, 137], les auteurs introduisent des distributions avec valuations qui donnent plus de poids aux bases non-réduites. Avec ce modèle plus réaliste, ils effectuent une analyse en moyenne de plusieurs paramètres dont les coûts additifs. Ils montrent par exemple que le nombre d'itérations K admet une queue de distribution géométrique qui dépend de la valuation.

Pour $d \geq 3$, Akhavi, Marckert et Rouault [24, 26] ont étudié les propriétés d'une base aléatoire dans le modèle sphérique (la distribution est invariante par rotation). Ils ont montré que si la dimension d du réseau et la dimension n de l'espace ambiant vérifient $n - d \to \infty$, alors une base aléatoire est déjà presque réduite avec probabilité proche de 1. Si au contraire $n - d$ reste constant, alors la probabilité tend vers une constante différente de 1.

Dans [43], Daudé et Vallée ont donné une borne supérieure sur le nombre moyen d'itérations de LLL avec $\tau < 1$ et avec le modèle de la boule uniforme (qui est un cas particulier de distribution sphérique). Pour un réseau de dimension maximale $d = n$, il est aussi connu que K admet une queue de distribution géométrique

$$\mathbb{P}[K \geq k] \leq \left(2\sqrt{n}\tau^{-1/n}\right)\tau^{k/n^2}.$$

Les résultats pour $d = 1, 2$ sont précis et assez complets avec des analyses en distribution. Pour $d \geq 3$, les résultats sont moins précis avec uniquement des bornes pour le nombre moyen d'itérations. De plus, le modèle sphérique ou le modèle de la boule uniforme ne correspondent pas à des situations pratiques où les réseaux sont mal réduits. Notre approche consiste à modéliser l'algorithme LLL pour en obtenir une version simplifiée. Nous proposons aussi des modèles aléatoires de réseaux, à valuations, qui sont capables de modéliser les réseaux difficiles à réduire. Nous analysons dans ces modèles aléatoires, nos modèles simplifiés de LLL.

3.2 Modélisations de LLL et des réseaux

La dynamique de l'algorithme LLL est complexe et nous ne parvenons pas à en réaliser une analyse probabiliste fine. Dans cette section, nous présentons des systèmes dynamiques qui modélisent des exécutions de l'algorithme LLL et que nous analysons à la section 3.3. Ensuite, nous présentons des modèles aléatoires de réseaux euclidiens suffisamment riches pour modéliser des applications réelles.

3.2.1 Modélisations de LLL

L'algorithme LLL effectue des translations et des échanges. Les translations sont réalisées entre chaque échange afin de proprifier la base. Les échanges visent à améliorer la base orthogonale et satisfaire les conditions de Lovász. Ces dernières ne dépendent que des coefficients sous diagonaux $\mathbf{m} = (m_{i+1,i})_{i=1..d-1}$ de la matrice de Gram-Schmidt et de la norme des orthogonalisés à travers les rapport $\mathbf{x} = (x_i)_{i=1..d-1}$. L'idée principale des modèles est de décrire l'évolution de LLL uniquement à travers les deux vecteurs $\mathbf{m}$ et $\mathbf{x}$.

Les règles de mise à jour de $\mathbf{x}$ et $\mathbf{m}$ après chaque échange sont décrites par les équations (3.2) et (3.3) et dépendent du facteur de décroissance $\rho = x_i + m_{i+1,i}^2$. Bien entendu, ce facteur évolue avec l'exécution de l'algorithme. Des translations réalisées à certaines étapes peuvent influer sur des échanges entre deux vecteurs de nombreuses étapes plus tard. Les modèles suppriment cette dépendance en simplifiant les règles de mise à jour. Nous présentons 3 modèles simplifiés.

Le modèle M1 des tas de sable. Le modèle $M_1(\tau, \rho)$ a été proposé par Madritsch et Vallée [98]. Dans ce modèle τ et ρ sont des constantes positives avec $\tau < 1$ et $\rho < 1$. Les conditions de Lovász sont remplacées par les conditions de Siegel (plus fortes), pour $i = 1..d-1$,

$$\mathcal{S}_\tau(i) : \qquad x_i \geq \tau.$$

Si la condition de Siegel à l'indice i n'est pas vérifiée, alors le vecteur $\mathbf{x}$ est remis à jour et le nouveau vecteur $\check{\mathbf{x}}$ vérifie

$$\check{x}_{i-1} = \rho \cdot x_{i-1}, \qquad \check{x}_i = \frac{1}{\rho^2} \cdot x_i, \qquad \check{x}_{i+1} = \rho \cdot x_{i+1}.$$

L'algorithme s'arrête dès que toutes les conditions de Siegel sont vérifiées.

A une transformation près, ce système dynamique est bien connu. Posons $y_i = \log_\tau x_i$. Alors, les conditions de Siegel s'écrivent

$$\mathcal{S}_\tau(i) : \qquad y_i \leq 1$$

puisque $\tau < 1$, et la règle de mise à jour devient

$$\check{y}_{i-1} = y_{i-1} + \alpha, \qquad \check{y}_i = y_i - 2\alpha, \qquad \check{y}_{i+1} = y_{i+1} + \alpha$$

avec $\alpha = \log_\tau \rho > 0$. Si l'on voit les y_i comme des tas de sable, cela revient à chaque étape à prendre une quantité de sable sur un tas qui en a trop, et à en verser une moité sur les deux tas à côté. Lorsque le sable tombe du bord, il est définitivement perdu. Les systèmes dynamiques de type tas de sable ou chip firing game (CFG) ont beaucoup de propriétés connues. En particulier, le nombre d'itérations et la configuration de sortie ne dépendent pas de la manière dont les tas sont choisis à chaque itération (appelée la stratégie). Ce modèle est très simple mais il a permis d'expliquer certains phénomènes qualitatifs de LLL. De plus, il a aussi été utilisé pour l'analyse d'un autre algorithme de réduction de réseau appelé BKZ [73]. La figure 3.2 montre une exécution d'un CFG avec $\alpha = 1$ et à valeurs entières.

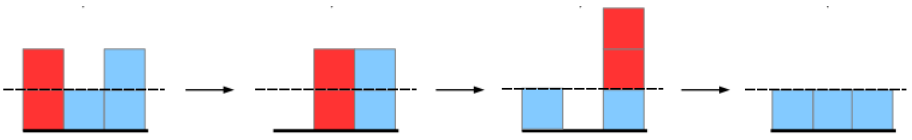


FIGURE 3.2 – Exemple d'exécution d'un CFG à valeurs entières avec $\alpha = 1$. A chaque étape, un carré foncé (rouge) est disposé à gauche et à droite. Sur les bords, un des carrés est perdu.

Le modèle M2. Le modèle $M_2(\tau, \mu)$ a été introduit dans la thèse de Mariya Georgieva [64] et dépend de deux paramètres $\tau \in [0, 1]$ et $\mu \in [0, \frac{1}{4}]$. Dans ce modèle, le facteur de décroissance ρ n'est plus supposé constant. Il dépend des valeurs de x_i et $m^2_{i+1,i}$ mais l'objectif principal de l'algorithme est de contrôler les x_i. Le modèle $M_2(\tau, \mu)$ donne un rôle prédominant aux x_i et un rôle secondaire aux $m^2_{i+1,i}$. Ces derniers sont supposés constants pendant l'exécution de l'algorithme, de valeur μ. Dans ce contexte, les conditions de Lovász s'écrivent pour $i = 1..d-1$,

$$\overline{\mathcal{C}}_{\tau,\mu}(i) : \qquad x_i + \mu \geq \tau.$$

Les règles de mise à jour sont aussi modifiées et deviennent

$$\check{x}_{i-1} = x_{i-1}(x_i + \mu), \qquad \check{x}_i = \frac{x_i}{(x_i + \mu)^2}, \qquad \check{x}_{i+1} = x_{i+1}(x_i + \mu).$$

L'algorithme s'arrête dès que toutes les conditions de Lovász $\overline{\mathcal{C}}_{\tau,\mu}(i)$ sont vérifiées. Ces règles définissent un système dynamique sur l'ensemble I_μ^{d-1} avec $I_\mu = [0, 1/(4\mu)]$. Le système s'arrête dès que $\mathbf{x}$ appartient au trou $[\tau - \mu, 1/(4\mu)]^{d-1}$. Lorsque $\tau = 1$, l'analyse de l'algorithme LLL est incomplète. C'est aussi un cas particulier pour le système dynamique puisqu'il a un point fixe sur la bordure du trou en $\mathbf{x} = (1-\mu, \ldots, 1-\mu)$. La figure 3.3 donne un exemple d'exécution du modèle $M_2(1, \mu)$ avec $\mu = 0.1$, la stratégie gloutonne (voir un peu plus loin) et $d = 3$. Le système dynamique est de dimension $d - 1 = 2$.

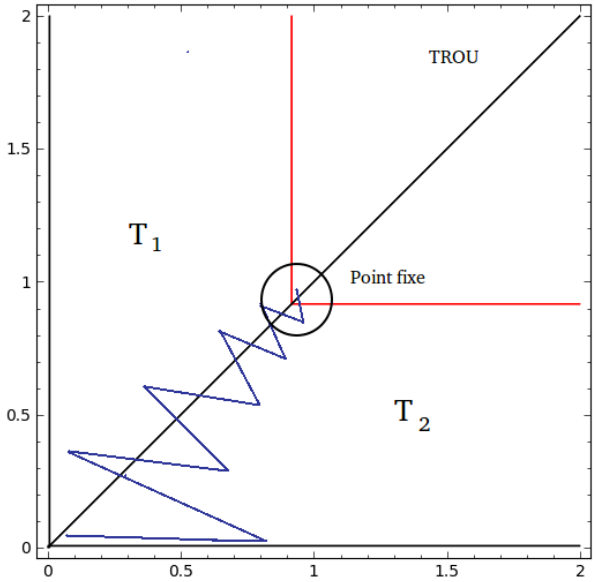


FIGURE 3.3 – Exemple d'exécution du système dynamique $M_2(1, \mu)$ avec $\mu = 0.1$, la stratégie gloutonne et $d = 3$. La zone T_1 est le domaine où la transformation s'applique en $i = 1$. De même pour T_2. Le point fixe en $(1 - \mu, 1 - \mu)$ est entouré. Le trou correspond au carré (rouge) en haut à droite.

Le modèle M3. Le modèle $M_3(\tau, f)$ a également été introduit dans la thèse de Mariya Georgieva. Il dépend du paramètre $\tau \in [0, 1]$ et d'une densité f sur l'intervalle $[0, 1/4]$. Dans ce modèle, les coefficients sous diagonaux $\mu_i = m^2_{i+1,i}$ ne sont plus supposés constants. A chaque

étape, ils sont tirés aléatoirement selon la densité de probabilité f. Dans ce contexte, les conditions de Lovász s'écrivent pour $i = 1..d-1$,

$$\widetilde{\mathcal{C}}_\tau(i) : \qquad x_i + \mu_i \geq \tau.$$

Les règles de mise à jour sont celles de l'algorithme LLL,

$$\check{x}_{i-1} = x_{i-1}(x_i + \mu_i), \qquad \check{x}_i = \frac{x_i}{(x_i+\mu_i)^2}, \qquad \check{x}_{i+1} = x_{i+1}(x_i + \mu_i)$$

auxquelles nous rajoutons la mise à jour de μ_i qui consiste à tirer une nouvelle valeur de μ_i selon la densité f. Ces règles définissent un système dynamique probabiliste sur l'ensemble $[0, +\infty[^{d-1} \times [0, 1/4]^{d-1}$. Le système s'arrête dès que $(x_1, \ldots, x_{d-1}, \mu_1, \ldots, \mu_{d-1})$ vérifie toutes les conditions de Lovász $\widetilde{\mathcal{C}}_\tau(i)$ pour $i = 1..d-1$.

Stratégies. Lorsqu'une base n'est pas τ-LLL-réduite, il existe plusieurs stratégies pour choisir l'indice i tel que la condition de Lovász $\mathcal{C}_\tau(i)$ n'est pas vérifiée. La stratégie classique de LLL consiste à choisir le plus petit entier i. La stratégie gloutonne choisit l'indice i tel que le facteur de décroissance est le plus favorable, c'est-a-dire $i = \text{argmin}\{x_i + m_{i+1,i}^2\ ,\ i = 1..d-1\}$. Dans la suite, nous supposons toujours que la stratégie gloutonne est utilisée.

3.2.2 Modèle aléatoire de réseaux euclidiens

Le modèle présenté dans ce chapitre, issu des travaux de thèse de Maryia Georgieva, s'inspire des distributions à valuations utilisées par les auteurs de [42] pour l'analyse en moyenne de l'algorithme de Gauss. Mariya a aussi exhibé de nombreuses applications que le modèle est capable de capturer. Ces applications liées à la cryptographie sont par exemple la factorisation de Schnorr, les réseaux de la méthode de Coppersmith pour attaquer le protocole RSA, les réseaux de Ajtaï pour les réductions pire cas/cas moyen, etc. Nous nous concentrons ici seulement sur le modèle et nous renvoyons à [64] pour les applications.

La difficulté de réduire un réseau $\mathcal{L}(\mathbf{b})$ « ne dépend pas vraiment » de la longueur des orthogonalisés $(||\mathbf{b}_1^\star||, \ldots, ||\mathbf{b}_d^\star||)$ puisque l'algorithme LLL effectuera les mêmes opérations avec le réseau $\mathcal{L}(\lambda\mathbf{b})$ pour un réel non nul λ. La difficulté dépend surtout des rapports des orthogonalisés $(x_1, \ldots, x_{d-1})$ avec $x_i = ||\mathbf{b}_{i+1}^\star||^2 / ||\mathbf{b}_i^\star||^2$ pour $i = 1..d-1$. Plus les x_i sont proches de 0, plus la base est mal réduite.

Le modèle dépend d'un vecteur de valuations $\mathbf{r} = (r_1, \ldots, r_{d-1})$ dont la composante r_i permet de contrôler la valeur moyenne de x_i.

Définition 1 (Modèle $\mathcal{M}_{\mathbf{r},a,d}$) *Considérons un vecteur de valuations $\mathbf{r} = (r_1, \ldots, r_d)$, un réel $a > 0$ et $d \in \mathbb{N}^\star$. Soit $f_\mathbf{r}$ la densité sur l'hypercube $H_a := [0, a]^{d-1}$ proportionnelle à $\mathbf{y}^{\mathbf{r}-1} := \prod_{i=1}^{d-1} y_i^{r_i - 1}$. Le modèle génère un réseau aléatoire $\mathcal{L}(\mathbf{b})$ dont les x_i vérifient $x_i \leq a^2$ et dont la densité de $y_i = x_i^{1/2}$ est f_r. La procédure est la suivante avec $(e_1, \ldots, e_d)$ la base canonique de $\mathbb{R}^d$:*

1. *Générer $\mathbf{y}$ selon la densité $f_\mathbf{r}$ sur H_a,*
2. *Générer uniformément dans $[-\frac{1}{2}, \frac{1}{2}]$ les coefficients $(m_{i,j})_{1 \leq j < i \leq d}$.*
3. *Calculer les vecteurs $\mathbf{b} = (b_1, \ldots, b_d)$ tels que pour $i = 1..d$*

$$b_i = \sum_{j=1}^{i} t_{i,j} e_j, \qquad t_{1,1} = 1, \qquad t_{i,i} = y_{i-1} t_{i-1,i-1},$$
$$\text{et pour } j = 1..i-1, \qquad t_{i,j} = m_{i,j} t_{j,j}.$$

La construction est faite de manière à ce que les coefficients de la partie triangulaire inférieure de la matrice M de Gram-Schmidt soient les $m_{i,j}$. Les vecteurs orthogonaux vérifient $b_i^\star = t_{i,i}e_i$ et satisfont $||b_{i+1}^\star||/||b_i^\star|| = y_i = x_i^{1/2}$.

3.3 Principaux résultats

La section 3.2 a introduit les modèles $M_1(\rho)$, $M_2(\tau,\mu)$ et $M_3(\tau,f)$ d'exécutions de l'algorithme LLL ainsi que le modèle probabiliste à valuations $\mathcal{M}_{\mathbf{r},a,d}$ de réseaux euclidiens. Nous décrivons dans cette section les premiers résultats d'analyses des modèles d'exécutions lorsque la distribution des entrées est donnée par $\mathcal{M}_{\mathbf{r},a,d}$.

3.3.1 Probabilité et potentiel de la base

Le premier résultat apporte des précisions sur le modèle $\mathcal{M}_{\mathbf{r},a,d}$ et dépend du potentiel $P(\mathbf{x})$ défini par

$$P(\mathbf{x}) := \prod_{i=1}^{d-1} x_i^{i(d-i)/2} = Q(\mathbf{b})^{-1}(\det \mathbf{b})^{d-1}. \tag{3.5}$$

Comme le déterminant est un invariant du réseau, $P(\mathbf{x})$ est bien un potentiel mais celui-ci croît d'un facteur $1/\rho$ après chaque échange avec ρ donné par l'équation (3.4).

Théorème 8 *Considérons le modèle $\mathcal{M}_{\mathbf{r},a,d}$ et posons $D = d(d^2-1)/6$. Soit $\mathbf{s}$ le vecteur dont les composantes s_i vérifient $s_i = r_i/(i(d-i))$. Nous notons $m(\mathbf{s})$ la plus petite composante de $\mathbf{s}$. Considérons maintenant l'ensemble $\mathcal{B}$ des indices i tels que $i := \operatorname{argmin}\{s_j\}$ et posons $b = \operatorname{card}\mathcal{B}$. Alors la distribution du potentiel $P(\mathbf{x})$ vérifie, pour tout $C \in [0,1]$,*

$$\mathbb{P}\left[P(\mathbf{x}) \le Ca^D\right] = \Theta\left(C^{m(\mathbf{s})}(\ln C)^{b-1}\right), \qquad C \to 0,$$

où les constantes dans le Θ ne dépendent que de $\mathbf{s}$. Lorsque les composantes de $\mathbf{s}$ sont toutes distinctes, le vecteur $\mathbf{r}$ est dit irréductible. Si $\mathbf{r}$ est irréductible, il existe une expression exacte, faisant intervenir le polynôme unitaire $S(x)$ dont les racines sont les composantes s_i, et donnée par

$$\mathbb{P}\left[P(\mathbf{x}) \le Ca^D\right] = -S(0)\sum_{i=1}^{d-1}\frac{C^{s_i}}{s_i S'(s_i)}.$$

Ce résultat est fondamental pour étudier le nombre K d'itérations des modèles puisque nous relions les évènements $[K \ge k]$ à des évènements de la forme $[P(\mathbf{x}) \le A(\mathbf{r},\tau,\mu)\mu^{km(\mathbf{s})}]$ avec $A(\mathbf{r},\tau,\mu)$ une constante qui ne dépend que du triplet $(\mathbf{r},\tau,\mu)$.

3.3.2 Nombre d'itérations du modèle $\mathbf{M_2}$

Nous avons étudié la distribution du nombre d'itérations du modèle $M_2(\tau,\mu)$ pour $\mu > 0$ et $\tau < 1$. Nous avons échoué à étudier le cas limite $\tau = 1$ sauf en dimensions $d = 2$ et $d = 3$. Les résultats sont résumés par la proposition suivante.

Proposition 4 *Considérons le modèle $\mathcal{M}_{\mathbf{r},a,d}$ dont le vecteur $\mathbf{s}$ associé est irréductible. Pour tout $\mu > 0$ et tout $\tau < 1$, le nombre d'itérations de l'algorithme $M_2(\tau,\mu)$ est asymptotiquement géométrique, et vérifie*

$$\mathbb{P}[K \ge k] = \Theta\left(\mu^{km(\mathbf{s})}\right), \qquad m(\mathbf{s}) := \min_{i=1..d-1}\frac{r_i}{i(d-i)}, \qquad k \to \infty.$$

Les constantes cachées dans le Θ dépendent de μ et τ et sont uniformes pour $\mu \in [\mu_0, 1/2]$ et $\tau \in [0, \tau_0]$ avec $\mu_0 > 0$ et $\tau_0 < 1$.
Les mêmes résultats s'appliquent pour $d = 2, 3$ et $\tau = 1$.

La loi géométrique est connue pour l'algorithme LLL lorsque $d = 2$ et $\tau = 1$. En effet, il s'agit de l'algorithme de Gauss dont l'analyse du nombre d'itérations peut être trouvée dans [42]. La proposition 4 confirme l'intuition que ce résultat s'étend à toutes les dimensions.

Nous n'avons pas traité le cas $\tau = 1$ en grandes dimensions car nous ne sommes pas parvenus à analyser la fin de l'algorithme lorsque toutes les entrées sont presque réduites. Lorsque $\tau = 1$, la fin de l'algorithme est ralentie par le point fixe du système dynamique. Il reste à comprendre comment l'algorithme contourne le point fixe pour terminer dans le trou.

3.3.3 Nombre d'itérations du modèle M_3

Le système dynamique $M_3(\tau, f)$ est un système probabiliste. Il y a donc deux distributions de probabilité qui co-existent : celle définie par le modèle des entrées $\mathcal{M}_{\mathbf{r},a,d}$ et celle induite par les trajectoires probabilistes du système. Nous ne sommes pas parvenus à obtenir un résultat similaire à la proposition 4. Nous avons toutefois obtenu le résultat suivant qui ne prend pas en compte le modèle des entrées.

Proposition 5 *Soit $\tau < 1$. Supposons que f est une densité sur $[\mu_0, 1/4]$ avec $\mu_0 > 0$. Si μ est une variable aléatoire de densité f, nous posons $\alpha = \mathbb{E}[\log \mu]$ et $\sigma^2 = \mathbb{V}(\log \mu)$ l'espérance et la variance de $\log \mu$. Soit $\epsilon > 0$. Si $d \to \infty$ et $P(\mathbf{x}) \to 0$ tels que $d^3 = o(\log P(\mathbf{x}))$, alors*

$$\mathbb{P}\left[\left|K \cdot \frac{\alpha}{\log P(\mathbf{x})} - 1\right| > \epsilon\right] \to 0.$$

Dans le modèle M_2, la preuve de la proposition 4 utilise un résultat intermédiaire similaire à la proposition 5. Ce résultat indique que le nombre d'itérations $K(\mathbf{x})$ est équivalent à $\log_\mu P(\mathbf{x})$ lorsque $P(\mathbf{x}) \to 0$. La proposition 5 remplace juste la constante $\log \mu$ du modèle $M_2(\tau, \mu)$ par la valeur moyenne de $\ln \mu$ avec μ la variable aléatoire de densité f.

Si $\tau = 1$, une difficulté supplémentaire apparait que nous ne retrouvons pas dans l'algorithme LLL.

Proposition 6 *Dans le modèle $M_3(1, f)$, si f est une densité continue sur $[\mu_0, 1/4]$ avec $\mu_0 > 0$, alors il existe des trajectoires infinies quelle que soit la dimension.*

Afin d'aller plus loin dans le modèle M_3, nous devons obtenir des résultats plus précis que la proposition 5 et nous devons borner la probabilité d'avoir des trajectoires très longues.

3.4 Conclusion

L'algorithme LLL a une dynamique trop complexe pour être étudiée directement. Dans ce chapitre, nous avons introduit des modèles simplifiés de l'exécution de LLL. Ces modélisations se concentrent sur les éléments essentiels de l'algorithme : les coefficients sous-diagonaux de la matrice de Gram-Schmidt et la norme des orthogonalisés. Nous avons commencé les analyses des modèles M_2 et M_3. Beaucoup de travail reste à faire sur le modèle M_3 où peu de résultats ont été obtenus. En revanche, nous avons obtenu des lois géométriques pour le nombre d'itérations dans le modèle M_2. Ce résultat est conforme à ce qui a été démontré en petite dimension avec l'algorithme de Gauss.

Le modèle M_2 n'a pas été analysé pour $\tau = 1$ en dimension $d \geq 4$. Les techniques employées pour $d = 3$ ne semblent pas se généraliser à cause d'une combinatoire trop grande des trajectoires autour du point fixe. Quasiment tout reste à faire sur le modèle M_3 qui se veut assez proche de LLL. Pour aller plus loin, il serait aussi intéressant d'étudier d'autres algorithmes de réduction de réseaux avec le point de vue des tas de sable.

Chapitre 4

Sources, taux d'entropie renormalisés et mots

Ce chapitre décrit les principaux résultats des articles suivants :
- [3] **Computation and estimation of generalized entropy rates for denumerable Markov chains.**
  G. Ciuperca, V. Girardin, L. Lhote.
  IEEE Transactions on Information Theory, 57(7) :4026–4034, 2011.
- [5] **Rescaling entropy and divergence rates.**
  V. Girardin, L. Lhote.
  IEEE Transactions on Information Theory, 61(11) :5868–5882, 2015.
- [9] **Toward the asymptotic count of bi-modular hidden patterns under probabilistic dynamical sources : a case study.**
  L. Lhote, M. Lladser
  Discrete Mathematics and Theoretical Computer Science Proceedings, AQ : 425-452, 2012

Ce chapitre a pour objet d'étude principal les sources. Une source est un procédé probabiliste qui émet des symboles selon un temps discret sur un alphabet possiblement infini. La suite de symboles forme des mots dont les propriétés sont analysées.

Les sources sont des objets transverses aux travaux présentés dans ce mémoire. Au chapitre 2, nous avons vu la transformation de Gauss aussi appelée source des fractions continues. Intuitivement, les symboles émis à chaque itération sont les quotients qui apparaissent dans l'exécution de l'algorithme d'Euclide. La transformation de Gauss est un cas particulier de source dynamique. Les sources dynamiques sont décrites au chapitre 1 section 1.3.3 et sont utilisées au chapitre 3 pour faire de la combinatoire des mots. Au chapitre 5, nous étudions les propriétés probabilistes d'hypergraphes aléatoires générés à partir de sources de Bernoulli. Dans tous ces travaux, la source sert d'aléa pour construire des objets qui sont ensuite analysés.

Notre point de vue dans ce chapitre est légèrement différent. Nous étudions les mots produits par une source ainsi qu'une grandeur caractéristique appelée *taux d'entropie*. Dans [3], nous nous sommes intéressés au calcul et à l'estimation de taux d'entropie pour une classe d'entropies généralisées aussi appelées (h, φ)-entropies. Nous avons obtenu des formules explicites dès que la source vérifie une propriété dite de quasi-puissance. Sous certaines hypothèses simples, toutes les sources classiques (sources sans mémoire, chaîne de Markov) vérifient cette propriété. Nous avons obtenu un résultat assez *étonnant* : lorsque les taux d'entropie ne sont pas nuls ou infinis, ils sont égaux à une constante près à ceux de Shannon ou Rényi. Autrement dit, exceptés les taux

d'entropie de Shannon et Rényi, les autres taux d'entropie ont peu d'intérêt (sous l'hypothèse de quasi-puissance).

Dans [5], nous avons souhaité corriger ce problème en donnant un sens aux (h, φ)-taux d'entropie. Nous nous sommes aperçus que le problème venait de la définition classique des taux d'entropie qui font intervenir une renormalisation en $1/n$. Cette renormalisation est bien adaptée pour les entropies de Shannon ou Rényi mais pas pour les autres. Nous avons alors étendu les travaux précédents dans deux directions : tout d'abord, nous avons élargi la notion de renormalisation. Ensuite, nous avons étudié les taux d'entropie relative ou taux de divergence. Les divergences mesurent *la distance* entre deux sources et l'entropie est un cas particulier de divergence où l'une des sources est la distribution uniforme.

En collaboration avec Manuel Lladser [9], nous nous sommes intéressés aux mots produits par une source. Nous avons étudié le nombre d'occurrences d'un motif caché dans un texte produit par une source dynamique. Ce paramètre est important pour les algorithmes de recherche de motifs et l'étude de la distribution des motifs trouve de nombreuses applications en biologie, en extraction de connaissances dans les textes, etc. Les motifs cachés ont déjà été analysés par les auteurs de [61] mais uniquement avec des sources sans mémoire. Nous avons considéré le cas des sources dynamiques mais nous nous sommes limités à des motifs extrêmement simples composés de deux lettres.

Plan. Ce chapitre est décomposé en deux sections. La première aborde les travaux sur le calcul des taux d'entropie généralisés (section 4.1). La section 4.1.1 présente les notions classiques de taux d'entropie ou de taux d'entropie relative avant d'introduire les taux d'entropie relative renormalisés qui sont au cœur de nos travaux. La section 4.1.2 décrit les hypothèses (quasi-puissance, quasi-poly-log) que nous faisons sur les sources et les fonctions qui définissent les entropies relatives. Nous énonçons notre principal résultat à la section 4.1.3 avant de l'appliquer à quelques exemples à la section 4.1.4. Finalement, nous présentons brièvement les travaux sur les motifs cachés à la seconde section 4.2. Nous avons choisi de moins détailler ces travaux car ils constituent un début de collaboration et sont en un certain sens moins aboutis.

4.1 Calcul explicite de taux d'entropie généralisés et renormalisés

Les travaux de cette section sont le fruit d'une collaboration commencée en 2007 avec Valérie Girardin. Depuis cette date, un groupe de travail portant sur les entropies et les mots réunit régulièrement des mathématiciens et des informaticiens de Caen ou Paris. Les échanges au cours de ce groupe de travail ont notamment conduit aux deux publications [3, 5].

4.1.1 Contexte

Dans cette section, $\mathcal{A}$ est un ensemble fini ou dénombrable appelé alphabet. Nous considérons deux sources, notées $\mathcal{S}$ et $\mathcal{S}'$, sur l'alphabet $\mathcal{A}$. L'ensemble $\mathcal{A}^\star = \bigoplus_{n\geq 0} \mathcal{A}^n$ désigne l'ensemble des mots finis construits sur l'alphabet $\mathcal{A}$. Les probabilités $p_{\mathbf{w}}$ et $p'_{\mathbf{w}}$ désignent respectivement les probabilités que les sources $\mathcal{S}$ et $\mathcal{S}'$ émettent un mot commençant par $\mathbf{w} \in \mathcal{A}^\star$.

4.1.1.1 Notions de taux d'entropie

Dans [123], Shannon a adapté au contexte de la théorie de l'information la notion d'entropie qui avait été introduite par Boltzman au 19ème siècle. L'entropie mesure l'aléa ou l'incertitude

$h(z)$	$\varphi(x)$	$(h,\varphi)-$ entropies
z	$-x\log x$	Shannon (1948)
$(1-s)^{-1}\log z$	x^s	Renyi (1961)
$[t(t-r)]^{-1}\log z$	$x^{r/t}$	Varma (1966)
z	$(1-2^{1-r})^{-1}(x-x^r)$	Havrda and Charvat (1967)
$(t-1)^{-1}(z^t-1)$	$x^{1/t}$	Arimoto (1971)
$(t-1)^{-1}[z^{(t-1)/(s-1)}-1]$	x^s	Sharma and Mittal 1 (1975)
$(r-1)^{-1}[\exp(r-1)z-1]$	$-x\log x$	Sharma and Mittal 2 (1975)
z	$-x^r\log x$	Taneja (1975)
z	$(t-r)^{-1}(x^r-x^t)$	Sharma and Taneja (1975)
$(r-1)^{-1}(1-z)$	x^r	Tsallis (1988)

TABLE 4.1 – Exemples de (h,φ)-entropies.

d'une source. Le taux d'entropie de Shannon de la source $\mathcal{S}$ est défini par la limite, si elle existe,

$$\mathbb{S}(\mathcal{S}) = -\lim_{n\to\infty}\frac{1}{n}\sum_{\mathbf{w}\in\mathcal{A}^n} p_{\mathbf{w}}\ln p_{\mathbf{w}}. \tag{4.1}$$

L'entropie de Shannon est maintenant appliquée à de nombreux domaines [40] comme la finance, la cryptographie, la physique, etc. Rényi fut le premier à proposer une extension de l'entropie de Shannon en introduisant des entropies paramétrées [114]. Le taux d'entropie de Rényi de $\mathcal{S}$ est défini par la limite, si elle existe,

$$\mathbb{R}_s(\mathcal{S}) = \lim_{n\to\infty}\frac{1}{n}\frac{1}{1-s}\ln\left(\sum_{\mathbf{w}\in\mathcal{A}^n} p_{\mathbf{w}}^s\right), \tag{4.2}$$

avec $s>0$. Depuis, de nombreuses entropies ont été définies dans des contextes applicatifs très variés comme les entropies de Tsallis [124] ou Sharma-Mittal [128]. Il est probablement impossible d'en faire une liste exhaustive mais un grand nombre d'entre elles font partie de la famille des (h,φ)-entropies, introduites par Salicrù *et al* dans [115]. Pour deux fonctions h et φ bien faites, le (h,φ)-taux d'entropie de la source $\mathcal{S}$ est défini par la limite, si elle existe,

$$\mathbb{S}_{h(z),\varphi(x)}(\mathcal{S}) = \lim_{n\to\infty}\frac{1}{n}h\left[\sum_{\mathbf{w}\in\mathcal{A}^n}\varphi(p_{\mathbf{w}})\right]. \tag{4.3}$$

La table 4.1 donne une liste non exhaustive de $(h,\varphi)-$ entropies.

Avant nos travaux dans [3], les valeurs explicites de (h,φ)-taux d'entropie étaient peu nombreuses et ne concernaient que des sources sans mémoire ou des chaînes de Markov. Une source $\mathcal{S}$ est dite *sans mémoire* de paramètre $(p_w)_{w\in\mathcal{A}}$ si, à chaque étape, $\mathcal{S}$ émet un nouveau symbole selon la distribution $(p_w)_{w\in\mathcal{A}}$ et de manière indépendante des précédents symboles. Les taux d'entropie de Shannon et de Rényi d'une source sans mémoire sont bien connus et vérifient

$$\mathbb{S}(\mathcal{S}) = -\lambda'(1) = -\sum_{w\in\mathcal{A}} p_w\ln p_w, \qquad \mathbb{R}_s(\mathcal{S}) = \frac{1}{1-s}\ln\lambda(s) \quad \text{avec} \quad \lambda(s)=\sum_{w\in\mathcal{A}} p_w^s.$$

Les chaînes de Markov sont des sources qui modélisent des corrélations bornées entre les symboles. L'idée est que le futur symbole ne dépend que du présent ou d'une partie bornée du passé. Une chaîne de Markov d'ordre 1 est définie par une distribution initiale $(f_w)_{w\in\mathcal{A}}$ sur l'alphabet $\mathcal{A}$ et par une matrice de transition $P=(p_{w|v})_{v,w\in\mathcal{A}^2}$. Le premier symbole est tiré aléatoirement

selon la distribution initiale qui sert à initialiser le processus. Ensuite si v est le symbole qui vient d'être émis, le symbole suivant est émis selon la distribution $(p_{w|v})_{w\in\mathcal{A}}$. Pour une chaîne de Markov ergodique, le taux d'entropie de Shannon s'exprime en fonction de la mesure invariante $\pi = (\pi_w)_{w\in\mathcal{A}}$ de la chaîne $(\pi \cdot P = \pi)$ à travers la relation [123]

$$\mathbb{S}(\mathcal{S}) = -\sum_{v,w\in\mathcal{A}} \pi_v \cdot p_{w|v} \cdot \ln p_{w|v}. \tag{4.4}$$

Pour des processus de Markov à temps continu, des résultats similaires sont obtenus dans [51]. Rached *et al* ont prouvé dans [106] que le taux d'entropie de Rényi pour une chaîne de Markov à espace d'états fini est

$$\mathbb{R}_s(\mathcal{S}) = \frac{1}{1-s}\ln\lambda(s), \tag{4.5}$$

avec $\lambda(s)$ l'unique valeur propre dominante de la matrice perturbée $P_s = (p^s_{w|v})_{v,w\in E}$ pour tout $s > 0$. Golshani *et al* ont traité le cas infini dénombrable dans [67] mais la preuve est incomplète. En faisant tendre s vers 1 dans (4.5), le taux d'entropie de Shannon est aussi relié à $\lambda(s)$ via la formule $\mathbb{S}(\mathcal{S}) = -\lambda'(1)$.

Les sources sans mémoire et les chaînes de Markov peuvent être décrites sous forme de sources dynamiques [132, 37], introduites au chapitre 1 section 1.3.3. Le taux d'entropie de Shannon d'une source dynamique s'exprime en fonction de l'unique valeur propre dominante $\lambda(s)$ des opérateurs de transfert associés au système à travers la relation $\mathbb{S}(\mathcal{S}) = -\lambda'(1)$. Les opérateurs de transfert pour les systèmes dynamiques jouent donc le même rôle que les matrices perturbées pour les chaînes de Markov.

Finalement, le taux d'entropie de Shannon de processus semi-markoviens à temps discret ou continu a été obtenu par Girardin et Limnios dans [65, 66].

Jusqu'à nos travaux en 2011 [3], ces valeurs explicites étaient les seules connues pour les taux d'entropie. Nous avons étendu les travaux existants aux (h,φ)-taux d'entropie mais aussi à toute source satisfaisant une propriété classique de quasi-puissance. Dans [5], nous avons encore étendu les travaux en étudiant les entropies relatives.

4.1.1.2 Notions de taux d'entropie relative

Les entropies relatives, aussi appelées divergences, mesurent la dissimilarité de deux sources. Dans [88], Kullback et Leibler introduisent le concept de divergence en théorie de l'information en proposant une *mesure* qui étend l'entropie de Shannon. Le taux d'entropie relative de Kullback-Leibler entre deux sources $\mathcal{S}$ et $\mathcal{S}'$ est défini par la limite, si elle existe,

$$\mathbb{K}(\mathcal{S}'/\mathcal{S}) = \lim_{n\to\infty}\frac{1}{n}\sum_{\mathbf{w}\in\mathcal{A}^n} p_{\mathbf{w}}\ln\frac{p_{\mathbf{w}}}{p'_{\mathbf{w}}}. \tag{4.6}$$

Comme pour les entropies, l'entropie relative de Kullback-Leibler a été étendue en de nombreuses divergences ayant des propriétés distinctes selon le contexte applicatif (voir par exemples [30, 127, 102, 129]). La plupart de ces entropies relatives sont des (h,φ)-divergences dont le taux est défini par la limite, si elle existe,

$$\mathbb{D}_{h(z),\varphi(x,y)}(\mathcal{S}'/\mathcal{S}) = \lim_{n\to\infty}\frac{1}{n}h\left(\sum_{\mathbf{w}\in\mathcal{A}^n}\varphi(p_{\mathbf{w}},p'_{\mathbf{w}})\right).$$

La table 4.2 recense plusieurs exemples de (h,φ)-divergences. Notez bien que dans le contexte des divergences, φ est une fonction à deux variables. En fait, si $\varphi(x,y) = \tilde{\varphi}(x)$ ne dépend que d'une

variable, le taux d'entropie relative est un taux d'entropie. Les taux d'entropie apparaissent alors comme un cas particulier de taux d'entropie relative. Nous nous limitons dans la suite à l'étude des entropies relatives.

Comme pour les taux d'entropie, il existe peu de résultats sur la valeur explicite des taux d'entropie relative. Si $\mathcal{S}$ et $\mathcal{S}'$ sont deux chaînes de Markov ergodiques à espaces d'états finis, le taux d'entropie de Kullback-Leibler vérifie

$$\mathbb{K}(\mathcal{S}'/\mathcal{S}) = \sum_{v,w\in\mathcal{A}} \pi_v p_{w|v} \ln \frac{p_{w|v}}{p'_{w|v}},$$

avec $P = (p_{w|v})_{v,w\in\mathcal{A}}$ et $P' = (p'_{w|v})_{v,w\in\mathcal{A}}$ les matrices de transition de $\mathcal{S}$ et $\mathcal{S}'$ et $(\pi_v)_{v\in\mathcal{A}}$ la mesure invariante de P. Dans [107], les auteurs ont montré que l'entropie relative de Rényi entre deux chaînes de Markov finies (et ergodiques) s'exprime en fonction de la valeur propre dominante d'une matrice qui mélange les deux matrices de transition. Les mêmes auteurs ont ensuite étendu le résultat à la divergence de Kullback-Leibler [108]. Kesidis et Walrand [82] ont déterminé l'entropie relative de Kullback-Leibler de deux processus Markoviens et Girardin et Limnios ont fait de même avec des processus semi-markoviens [65, 66].

A notre connaissance et jusqu'à nos travaux en 2015 [5], ces valeurs explicites étaient les seules connues pour les (h, φ)-taux d'entropie relative.

4.1.1.3 Taux d'entropie relative renormalisés

Dans [3], nous avons calculé le (h, φ)-taux d'entropie, défini par la formule (4.3). Nous avons obtenu des expressions explicites lorsque les fonctions h et φ sont dites *quasi-poly-log* et lorsque la source suit une hypothèse de quasi-puissance. La fonction univariée $\varphi(x)$ est quasi-poly-log si elle vérifie l'équivalence

$$\varphi(x) \underset{x\to 0}{\sim} c \cdot x^{\alpha} \cdot (\log x)^k$$

pour $\alpha \in \mathbb{R}$, $c > 0$ et $k \in \mathbb{N}^*$. La source $\mathcal{S}$ vérifie l'hypothèse de quasi-puissance si les sommes $\sum_{\mathbf{w}\in\mathcal{A}^n} p_{\mathbf{w}}^s$ sont asymptotiquement des quasi-puissances c'est-à-dire équivalentes à $c(s)\lambda(s)^n$ avec $c(s)$ et $\lambda(s)$ des fonctions *bien faites*. Nous détaillerons ces propriétés plus tard en les adaptant au contexte des entropies relatives. Néanmoins, beaucoup de (h, φ)-entropies sont quasi-poly-log, y compris les plus connues comme celles de Shannon ou Rényi. Sous des hypothèses classiques, les sources sans-mémoire, les chaînes de Markov à espace d'états fini ou dénombrable et les sources dynamiques satisfont aussi la propriété de quasi-puissance. Nos résultats s'appliquent de fait à une large classe de sources et de (h, φ)-entropies. Le résultat principal de [3] se résume en le théorème suivant.

Théorème 9 *Soit h et φ deux fonctions à valeurs réelles et quasi-poly-log. Soit $\mathcal{S}$ une source qui vérifie la propriété de quasi-puissance. Alors le (h, φ)-taux d'entropie de la source $\mathcal{S}$, défini par la formule* (4.3)*, est soit nul, soit infini, soit égal à une constante près (qui ne dépend que de (h, φ) et non $\mathcal{S}$) au taux d'entropie de Shannon ou de Rényi de $\mathcal{S}$.*

Le théorème 9 constitue un frein majeur à l'utilisation des (h, φ)-taux d'entropie dans le cadre assez classique des sources satisfaisant la propriété de quasi-puissance. Cette situation est due à la combinaison de deux faits : la propriété de quasi-puissance qui entraine une croissance ou une décroissance exponentielle des sommes $\sum_{\mathbf{w}\in\mathcal{A}^n} p_{\mathbf{w}}^s$ et la renormalisation classique en $1/n$ dans la définition 4.3 qui n'est pas toujours adaptée. Les entropies de Shannon et Rényi font intervenir la fonction logarithme qui, avec les sommes précédentes, fait apparaître un terme

	$h(z)$	$\varphi(x,y)$	$(h,\varphi)-$ divergences
1		poly-log :	
	z	$x\log(x/y)$	Kullback-Leibler
	z	$(y/x)-\ln(y/x)-1$	Itakura-Saito
2	poly-log ou poly :	poly :	
	z	$(x-y)^2/y$	χ^2
	$\frac{1}{1-s}\log z$	$y(x/y)^s$	Rényi
	$z^{1/s}$	$y^{2s}-x^s y^s$	Matusita
	$z^{1/s}$	$x^{1/s}-y^{1/s}$	Matusita
	z	$y^{1+s}-(1+1/s)xy^s+x^{1+s}/s$	Basu-Harris-Hjort-Jones
3	quasi-poly-log :	poly-log :	
	$\frac{1}{1-q}(z-1)$	$y(x/y)^q$	Tsallis-Harvda-Charvat
	z	$x\ln_{q^*}(x/y)$	dual Tsallis
	$\frac{1}{r-1}[z^{\frac{r-1}{s-1}}-1]$	$y(x/y)^s$	Sharma-Mittal 1
	$\frac{1}{1-s}\left[\left(1+\frac{1-r}{1-s}\log z\right)^{\frac{1-s}{1-r}}-1\right]$	$y(x/y)^s$	Masi
	$\frac{1}{r-1}[\exp(r-1)z-1]$	$x\log(x/y)$	Sharma-Mittal 2
4		quasi-poly :	
	$s^{-1}z$	$(1+sx)\log[(1+sx)/(1+sy)]$	Ferreri
5		pas quasi-poly-log :	
	z	$y(x-y)^2/(x+y)^2$	Balakrishnan-Sanghvi
	z	$(q-1)^{-1}x(x^{q-1}-y^{q-1})-(x-y)^{q-1}$	Venkatesan
	$2^{-1/t}z$	$(x^t+y^t)^{1/t}$	Moyenne harmonique

TABLE 4.2 – Exemples de (h,φ)-divergences, classées selon la nature de h et φ.

linéaire en n. La renormalisation en $1/n$ est alors bien adaptée. Si le logarithme disparait (c'est la cas dans beaucoup de (h,φ)-entropies), alors la normalisation en $1/n$ s'applique à des sommes exponentielles, ce qui aboutit à des valeurs triviales comme 0 ou $+\infty$.

Afin de donner un sens aux (h,φ)-taux d'entropie (relative), nous avons introduit les (h,φ)-taux d'entropie (relative) renormalisés [2].

Définition 2 *Soit $h(z)$ une fonction réelle univariée, $\varphi(x,y)$ une fonction réelle bivariée et $\mathbf{r} = (r_n)_n$ une suite de réels non nuls. Considérons deux sources $\mathcal{S}$ et $\mathcal{S}'$. Le $(h,\varphi,\mathbf{r})$-taux d'entropie relative renormalisé des sources $\mathcal{S}$ et $\mathcal{S}'$ est défini par la limite, si elle existe,*

$$\mathbb{D}_{h,\varphi,\mathbf{r}}(\mathcal{S}'/\mathcal{S}) = \lim_{n\to\infty} \frac{1}{r_n} h\left(\sum_{\mathbf{w}\in\mathcal{A}^n} \varphi(p_{\mathbf{w}}, p'_{\mathbf{w}})\right). \tag{4.7}$$

Si $\varphi(x,y)$ ne dépend que d'une variable (x ou y), alors $\mathbb{D}_{h,\varphi,\mathbf{r}}$ est appelé un $(h,\varphi,\mathbf{r})$-taux d'entropie renormalisé.

La renormalisation de taux d'entropie existe déjà dans plusieurs contextes comme les réseaux euclidiens et l'entropie *patch-counting* [89], les réseaux de Pétri [19], l'entropie de courbes planes finies [96], etc. Cependant à notre connaissance, les taux d'entropie relative renormalisés n'ont jamais été étudiés dans le cadre de la théorie de l'information.

4.1.2 Hypothèses et principaux résultats

Dans [5], nous avons calculé explicitement les taux d'entropie relative renormalisés. Le calcul s'appuie sur plusieurs hypothèses assez générales qui s'appliquent à la fois à de nombreuses sources et à de nombreuses entropies relatives. Nous commençons par énoncer les hypothèses sur les sources.

4.1.2.1 Hypothèses sur les sources

La première hypothèse pourrait s'énoncer de la manière suivante : plus un mot est long, plus sa probabilité d'être émis est faible et tend uniformément vers 0.

Propriété 2 (Source sans point d'accumulation) *Une source $\mathcal{S}$ est dite sans point d'accumulation si*

$$\sup_{\mathbf{w}\in\mathcal{A}^n} p_{\mathbf{w}} \underset{n\to\infty}{\to} 0.$$

Dans la suite, nous supposons toujours que les sources $\mathcal{S}$ et $\mathcal{S}'$ n'ont pas de point d'accumulation. Pour une source sans mémoire de paramètre $(p_w)_{w\in\mathcal{A}}$, cela signifie que la source n'est pas dégénérée ou que pour tout $w \in \mathcal{A}$, $p_w < 1$. Pour les chaînes de Markov à espace d'états fini ou dénombrable, il suffit qu'une puissance de la matrice de transition ait des coefficients strictement plus petits que $1-\epsilon$ pour un $\epsilon > 0$. Pour les sources dynamiques, cette propriété est assurée par un système dynamique dilatant ($|T'| > 1$ si T est la transformation). Sous des hypothèses simples, toutes les sources *classiques* sont sans point d'accumulation.

Nous présentons maintenant la propriété de quasi-puissance locale, introduite dans [5], qui est adaptée aux taux d'entropie relative renormalisés. Cette propriété étend la propriété de quasi-puissance utilisée dans [3] pour l'étude des (h,φ)-taux d'entropie. L'objet fondamental de

2. Ce sont les taux qui sont renormalisés et non les entropies. Ceci explique l'accord.

la propriété de quasi-puissance locale est la série de Dirichlet $\Lambda_n(\mathcal{S},\mathcal{S}';\mathbf{s};\mathbf{e})$ des probabilités des mots de longueur n définie par

$$\Lambda_n(\mathcal{S},\mathcal{S}';\mathbf{s};\mathbf{e}) := \sum_{\mathbf{w}\in\mathcal{A}^n} p_{\mathbf{w}}^{s_1}(p'_{\mathbf{w}})^{s_2}\big(\ln p_{\mathbf{w}}\big)^{e_1}\big(\ln p'_{\mathbf{w}}\big)^{e_2}. \tag{4.8}$$

avec $\mathbf{s}=(s_1,s_2)$ et $\mathbf{e}=(e_1,e_2)$. Les taux d'entropie s'expriment parfois directement avec la série de Dirichlet 4.8. Par exemple, le taux d'entropie de Kullback-Leibler donné par l'équation 4.6 s'écrit

$$\mathcal{K}(\mathcal{S}'/S)=\lim_{n\to\infty}\frac{1}{n}\left[\Lambda_n(\mathcal{S},\mathcal{S}';1,0;1,0)-\Lambda_n(\mathcal{S},\mathcal{S}';1,0;0,1)\right].$$

Le taux d'entropie relative de Rényi est défini par les fonctions $h(z)=\frac{1}{1-s}\ln z$ et $\varphi(x,y)=x^sy^{1-s}$ et est donné par la limite, si elle existe,

$$\mathbb{D}_{\frac{1}{1-s}\ln z,x^sy^{1-s},\mathbf{r}_n}(\mathcal{S}'/\mathcal{S})=\lim_{n\to\infty}\frac{1}{r_n}\frac{1}{1-s}\ln\Lambda_n(\mathcal{S},\mathcal{S}';s,1-s;0,0).$$

Nous pouvons multiplier les exemples de taux d'entropie (relative) qui font intervenir directement la série de Dirichlet $\Lambda_n(\mathcal{S},\mathcal{S}';\mathbf{s};\mathbf{e})$. La propriété de quasi-puissance locale dit que Λ_n se comporte localement comme une puissance d'une fonction analytique.

Propriété 3 (Quasi-puissance locale) *Deux sources $\mathcal{S}$ et $\mathcal{S}'$ satisfont la propriété de quasi-puissance locale au point $\mathbf{s}^\star=(s_1^\star,s_2^\star)\in\mathbb{R}^2$ s'il existe un voisinage complexe $\mathcal{V}$ de $\mathbf{s}^\star$, $\theta\in[0,1[$ et des fonctions analytiques λ, c et R_n sur $\mathcal{V}$ tels que pour tout $\mathbf{s}=(s_1,s_2)\in\mathcal{V}$ et tout $n\geq 0$,*

$$\Lambda_n(\mathcal{S},\mathcal{S}';\mathbf{s};0,0)=\lambda(\mathbf{s})^n c(\mathbf{s})+R_n(\mathbf{s}), \tag{4.9}$$

avec $|R_n(\mathbf{s})|\leq|\theta\lambda(\mathbf{s})|^n$, $\lambda(\mathcal{V}\cap\mathbb{R}^2)\subset\mathbb{R}^+$ et $c(\mathcal{V}\cap\mathbb{R}^2)\subset\mathbb{R}^+$.

Une fois de plus, cette propriété s'applique à de nombreuses sources *classiques*. Si $\mathcal{S}$ et $\mathcal{S}'$ sont deux sources sans-mémoire de paramètres $(p_w)_{w\in\mathcal{A}}$ et $(p'_w)_{w\in\mathcal{A}}$, alors

$$\Lambda_n(\mathcal{S},\mathcal{S}';\mathbf{s};0,0)=\lambda(\mathbf{s})^n \quad\text{avec}\quad \lambda(\mathbf{s})=\sum_{w\in\mathcal{A}}p_w^{s_1}(p'_w)^{s_2}.$$

La série $\Lambda_n(\mathcal{S},\mathcal{S}';\mathbf{s};0,0)$ est une *vraie* puissance et la propriété de quasi-puissance est vérifiée dans le domaine de convergence de la série $\lambda(\mathbf{s})$.

La proposition suivante montre que pour deux chaînes de Markov, une propriété de Perron-Frobenius combinée avec des conditions d'analyticité suffisent à obtenir la propriété de quasi-puissance locale.

Proposition 7 *Considérons deux chaînes de Markov $\mathcal{S}$ et $\mathcal{S}'$, de matrices de transition $P=(p_{w|v})_{v,w\in\mathcal{A}}$ et $P'=(p'_{w|v})_{v,w\in\mathcal{A}}$ et de densités initiales $\mathbf{f}=(f_w)_{w\in\mathcal{A}}$ et $\mathbf{f}'=(f'_w)_{w\in\mathcal{A}}$. Pour $\mathbf{s}\in\mathbb{R}^2$, nous posons $T(\mathbf{s})=\left(p_{w|v}^{s_1}(p'_{w|v})^{s_2}\right)_{v,w\in\mathcal{A}}$ et $\mathbf{g}(\mathbf{s})=\left(f_w^{s_1}(f'_w)^{s_2}\right)_{w\in\mathcal{A}}$. Les chaînes de Markov $\mathcal{S}$ et $\mathcal{S}'$ vérifient la propriété locale de quasi-puissance en $\mathbf{s}^\star=(s_1^\star,s_2^\star)$ dès que les hypothèses suivantes sont vérifiées :*

(a) Il existe un voisinage complexe $\mathcal{V}$ de $\mathbf{s}^\star$ tel que pour tout $\mathbf{s}\in\mathcal{V}$, l'opérateur $\mathbf{x}\mapsto\mathbf{x}\cdot T(\mathbf{s})$ est un opérateur borné sur $\ell^1(\mathbb{C},\mathcal{A})$ muni de la norme ℓ_1 ($\|(x_w)_{w\in\mathcal{A}}\|=\sum_{w\in\mathcal{A}}|x_w|$).

(b) L'application $\mathbf{s}\mapsto T(\mathbf{s})$ est analytique sur $\mathcal{V}$.

(c) La matrice $T(\mathbf{s}^\star)$ admet sur $\ell^1(\mathbb{C},\mathcal{A})$ une unique valeur propre simple de module maximum avec un saut spectral (propriété de Perron-Frobenius).

(d) Pour tout $\mathbf{s}\in\mathcal{V}$, $\mathbf{g}(\mathbf{s})$ appartient à $\ell^1(\mathbb{C},\mathcal{A})$ et $\mathbf{s}\mapsto\mathbf{g}(\mathbf{s})$ est analytique sur $\mathcal{V}$.

Dans [5], nous avons aussi donné des conditions suffisantes explicites pour que deux chaînes de Markov à espace d'états fini ou dénombrable satisfassent la propriété de quasi-puissance locale (voir Proposition 5 dans [5]). Il n'est pas utile d'entrer dans ces détails pour comprendre la suite du chapitre.

Si $T(\mathbf{s}^\star)$ vérifie la propriété de Perron-Frobenius, par analyticité, il en est de même de $T(\mathbf{s})$ dans un voisinage de $\mathbf{s}^\star$. La matrice $T(\mathbf{s})$ admet alors des vecteurs propres gauche et droite $l(\mathbf{s})$ et $r(\mathbf{s})$ associés à la valeur propre dominante $\lambda(\mathbf{s})$, et $T(\mathbf{s})$ vérifie la décomposition spectrale

$$T(\mathbf{s})^n = \lambda(\mathbf{s})^n \frac{r(\mathbf{s}) \cdot l(\mathbf{s})}{(l(s) \cdot r(s))} + R(\mathbf{s})^n$$

avec $(u \cdot v)$ le produit scalaire de u et v, et $R(\mathbf{s})$ le projecteur sur le reste du spectre. En combinant la décomposition spectrale avec la relation

$$\Lambda_n(\mathcal{S}, \mathcal{S}'; \mathbf{s}; 0, 0) = \mathbf{g}(\mathbf{s}) \cdot T(\mathbf{s})^{n-1} \cdot \mathbf{1},$$

il est clair que la propriété de Perron-Frobenius entraine la propriété de quasi-puissance locale pour des chaînes de Markov. La propriété de quasi-puissance locale donne l'asymptotique des séries $\Lambda_n(\mathcal{S}, \mathcal{S}'; \mathbf{s}; 0, 0)$, mais qu'en est-il des séries $\Lambda_n(\mathcal{S}, \mathcal{S}'; \mathbf{s}; \mathbf{e})$ avec $\mathbf{e} = (e_1, e_2) \in \mathbb{N}^2$? A l'aide des propriétés analytiques, il suffit de dériver par rapport à s_1 ou s_2 l'expression de $\Lambda_n(\mathcal{S}, \mathcal{S}'; \mathbf{s}; 0, 0)$ pour obtenir la proposition fondamentale suivante.

Proposition 8 *Soit $\mathcal{S}$ et $\mathcal{S}'$ deux sources sans point d'accumulation qui satisfont la propriété de quasi-puissance en $\mathbf{s}^\star = (s_1^\star, s_2^\star)$. Alors pour $\mathbf{s}$ dans un voisinage complexe de $\mathbf{s}^\star$ et pour $\mathbf{e} = (e_1, e_2) \in \mathbb{N}^2$, la série $\Lambda_n(\mathcal{S}, \mathcal{S}'; \mathbf{s}; \mathbf{e})$ vérifie*

$$\Lambda_n(\mathcal{S}, \mathcal{S}'; \mathbf{s}; \mathbf{e}) = c(\mathbf{s}) \cdot n^{e_1+e_2} \cdot \lambda'_{s_1}(\mathbf{s})^{e_1} \cdot \lambda'_{s_2}(\mathbf{s})^{e_2} \cdot \lambda(\mathbf{s})^{n-e_1-e_2} \left(1 + O\left(\frac{1}{n}\right)\right). \tag{4.10}$$

La constante dans le O est uniforme pour $\mathbf{s}$ dans le voisinage complexe de $\mathbf{s}^\star$.

Notez que l'hypothèse *sans point d'accumulation* combinée avec la propriété de quasi-puissance entrainent que les dérivées $\lambda'_{s_1}(\mathbf{s})$ et $\lambda'_{s_2}(\mathbf{s})$ sont non nulles (voir Lemme 1 dans [5]).

4.1.2.2 Hypothèses sur les fonctions

Nos travaux se limitent à une classe assez générale de (h, φ)-taux d'entropie qui englobe une grande part des taux d'entropie (relative) classiques. Comme nous supposons que les sources $\mathcal{S}$ et $\mathcal{S}'$ sont sans point d'accumulation, nous sommes amenés à étudier la fonction φ autour de $(0, 0)$. Nous supposons qu'en ce point, φ est de type quasi-poly-log au sens de la propriété suivante.

Propriété 4 (φ de type quasi-poly-log) *Soit $\varphi(x, y)$ une fonction réelle bivariée. La fonction $\varphi(x, y)$ est dite de type quasi-poly-log en $(0, 0)$ si elle satisfait l'équivalence*[3]

$$\varphi(x, y) \underset{(x,y)\to(0,0)}{\sim} \sum_{(\mathbf{s},\mathbf{e})\in E_\varphi} a_{(\mathbf{s},\mathbf{e})} x^{s_1} y^{s_2} \big(\ln x\big)^{e_1} \big(\ln y\big)^{e_2}, \tag{4.11}$$

avec E_φ un sous-ensemble fini de $\mathbb{R} \times \mathbb{R} \times \mathbb{N} \times \mathbb{N}$ et $a_{(\mathbf{s},\mathbf{e})}$ des réels non-nuls.

3. Nous écrivons $\varphi(x, y) \sim_{(x_0,y_0)} f(x, y)$ dès qu'il existe un voisinage $\mathcal{V}$ de (x_0, y_0) et une fonction ε définie sur $\mathcal{V}$ tels que $\varphi(x, y) = f(x, y)(1 + \varepsilon(x, y))$ avec $\lim_{(x,y)\to(x_0,y_0)} \varepsilon(x, y) = 0$.

En pratique, E_φ contient rarement plus de deux ou trois éléments. Par exemple dans le cas de la divergence de Kullback-Leibler, $\varphi(x,y) = x\ln(x/y)$ est de type quasi-poly-log, $E_\varphi = \{(1,0,1,0),(1,0,0,1)\}$ et $a_{(1,0,1,0)} = -a_{(1,0,0,1)} = 1$. Dans le cas de la divergence de Rényi, $\varphi(x,y) = x^s y^{1-s}$ est de type quasi-poly-log, $E_\varphi = \{(s,1-s,0,0)\}$ et $a_{(s,1-s,0,0)} = 1$.

Selon la définition (4.7), les taux d'entropie relative font intervenir la suite de sommes $\sum_{\mathbf{w}\in\mathcal{A}^n} \varphi(p_\mathbf{w}, p'_\mathbf{w})$ auxquelles s'applique la fonction h. Si nous notons z_0 la limite des sommes, le comportement de h autour de z_0 est essentiel pour le calcul explicite des taux d'entropie relative. Nous supposons que autour de z_0, h est de type quasi-poly-log au sens de la définition suivante.

Propriété 5 (h de type quasi-poly-log) *Soit $z_0 \in \mathbb{R}\cup\{\pm\infty\}$ et $h(z)$ une fonction réelle univariée. La fonction $h(z)$ est dite de type quasi-poly-log en z_0 si elle satisfait l'équivalence*

$$h(z) \underset{z\to z_0}{\sim} b\cdot z^t\cdot(\ln z)^u \quad \textit{avec} \quad (b,u,v)\in\mathbb{R}^\star\times\mathbb{R}\times\mathbb{R}. \tag{4.12}$$

Par exemple pour l'entropie relative de Kullback-Leibler, $h(z) = z$ est un polynôme et est de type quasi-poly-log. Pour la divergence de Rényi, $h(z) = \frac{1}{1-s}\ln z$ est un logarithme (à une constante près) et est aussi de type quasi-poly-log.

4.1.3 Valeurs explicites des taux d'entropie renormalisés

Nous disposons maintenant de quasiment toutes les notions nécessaires pour énoncer notre résultat principal mais commençons d'abord par un raisonnement mathématique non rigoureux.

Les taux d'entropie renormalisés sont donnés pour deux sources $\mathcal{S}$ et $\mathcal{S}'$ par la formule (4.7). Les taux font intervenir la somme $\sum_{\mathbf{w}\in\mathcal{A}^n}\varphi(p_\mathbf{w},p'_\mathbf{w})$ avec $(p_\mathbf{w},p'_\mathbf{w})\to 0$ lorsque $n\to\infty$. Si $\varphi(x,y)$ est de type quasi-poly-log en $(0,0)$, on a intuitivement l'équivalence

$$\sum_{\mathbf{w}\in\mathcal{A}^n}\varphi(p_\mathbf{w},p'_\mathbf{w}) \underset{n\to\infty}{\sim} \sum_{\mathbf{w}\in\mathcal{A}^n}\sum_{(\mathbf{s},\mathbf{e})\in E_\varphi} a_{(\mathbf{s},\mathbf{e})} p_\mathbf{w}^{s_1}\left(p'_\mathbf{w}\right)^{s_2}\left(\ln p_\mathbf{w}\right)^{e_1}\left(\ln p'_\mathbf{w}\right)^{e_2}.$$

En inversant les deux sommes, nous obtenons l'équivalence

$$\sum_{\mathbf{w}\in\mathcal{A}^n}\varphi(p_\mathbf{w},p'_\mathbf{w}) \underset{n\to\infty}{\sim} \sum_{(\mathbf{s},\mathbf{e})\in E_\varphi} a_{(\mathbf{s},\mathbf{e})}\Lambda_n(\mathcal{S},\mathcal{S}';\mathbf{s};\mathbf{e}). \tag{4.13}$$

Si pour tout $(\mathbf{s},\mathbf{e})\in E_\varphi$, les sources $\mathcal{S}$ et $\mathcal{S}'$ vérifient l'hypothèse de quasi-puissance locale en $\mathbf{s}$, alors intervertir les deux sommes ne pose pas de problème car une des sommes est finie et les séries sont uniformément convergentes. En revanche, des précautions doivent être prises pour la première équivalence pour éviter que des termes dominants ne s'annulent. Nous introduisons une relation d'ordre partielle sur l'ensemble E_φ et une propriété de non-dégénérescence.

Définition 3 (Ordre partiel sur E_φ) *Soit $\varphi(x,y)$ une fonction réelle bivariée de type quasi-poly-log en $(0,0)$ et considérons l'ensemble E_φ tel que défini dans la propriété 4. Soit $\mathcal{S}$ et $\mathcal{S}'$ deux sources telles que pour tout $(\mathbf{s},\mathbf{e})\in E_\varphi$, $\mathcal{S}$ et $\mathcal{S}'$ vérifient la propriété de quasi-puissance locale en $\mathbf{s}$. Nous conservons les notations des propriétés 3 et 4.*

Pour $(\mathbf{s},\mathbf{e})$ et $(\mathbf{s}',\mathbf{e}')$ deux éléments de E_φ, nous écrivons $(\mathbf{s},\mathbf{e})\succ(\mathbf{s}',\mathbf{e}')$ dès que $[\lambda(\mathbf{s})>\lambda(\mathbf{s}')]$ ou $[\lambda(\mathbf{s})=\lambda(\mathbf{s}')$ et $e_1+e_2>e'_1+e'_2]$. La relation « $\succ$ » définit un ordre partiel sur E_φ et nous notons E_φ^+ les éléments maximaux de E_φ pour la relation « $\succ$ ».

Contrairement aux propriétés 3 et 4, la propriété qui suit mêle à la fois les sources $\mathcal{S}$ et $\mathcal{S}'$ et la fonction φ.

Propriété 6 (Non dégénérescence des sommes) *Soit φ une fonction quasi-poly-log au sens de la propriété 4 et faisant intervenir l'ensemble E_φ. Soit $\mathcal{S}$ et $\mathcal{S}'$ deux sources telles que pour tout $(\mathbf{s},\mathbf{e}) \in E_\varphi$, $\mathcal{S}$ et $\mathcal{S}'$ vérifient la propriété de quasi-puissance locale en $\mathbf{s}$. Alors la somme $\sum_{\mathbf{w}\in\mathcal{A}^n} \varphi(p_{\mathbf{w}}, p'_{\mathbf{w}})$ est dite non-dégénérée dès que*

$$d(\mathcal{S},\mathcal{S}',\varphi) := \sum_{(\mathbf{s},\mathbf{e})\in E_\varphi^+} a_{(\mathbf{s},\mathbf{e})} \cdot c(\mathbf{s}) \cdot \lambda'_{s_1}(\mathbf{s})^{e_1} \cdot \lambda'_{s_2}(\mathbf{s})^{e_2} \neq 0. \tag{4.14}$$

Sous l'hypothèse de non dégénérescence des sommes, l'équivalence (4.13) est vérifiée et nous avons pour tout $(\mathbf{s},\mathbf{e}) \in E_\varphi^+$,

$$\sum_{\mathbf{w}\in\mathcal{A}^n} \varphi(p_{\mathbf{w}}, p'_{\mathbf{w}}) \underset{n\to\infty}{\sim} d(\mathcal{S},\mathcal{S}',\varphi) \cdot n^{e_1+e_2} \cdot \lambda(\mathbf{s})^{n-e_1-e_2}.$$

En supposant de plus que h est de type quasi-poly-log en un point z_0 donné par la limite de l'équivalence ci-dessus, nous obtenons notre principal résultat.

Théorème 10 *Soit φ une fonction quasi-poly-log au sens de la propriété 4 et faisant intervenir l'ensemble E_φ. Soit $\mathcal{S}$ et $\mathcal{S}'$ deux sources telles que pour tout $(\mathbf{s},\mathbf{e}) \in E_\varphi$, $\mathcal{S}$ et $\mathcal{S}'$ vérifient la propriété de quasi-puissance locale en $\mathbf{s}$. Supposons de plus que le triplet $(\mathcal{S},\mathcal{S}',\varphi)$ satisfait la propriété 6 de non dégénérescence des sommes. Pour $(\mathbf{s},\mathbf{e}) \in E_\varphi^+$, nous posons*

$$z_0 = \lim_{n\to\infty} d(\mathcal{S},\mathcal{S}',\varphi) \cdot n^{e_1+e_2} \cdot \lambda(\mathbf{s})^{n-e_1-e_2}, \tag{4.15}$$

avec $d(\mathcal{S},\mathcal{S}',\varphi)$ défini par l'équation (4.14). Supposons que h est de type quasi-poly-log en z_0 au sens de la propriété 5. Alors trois cas sont possibles.

(i) Si $\lambda(\mathbf{s}) \neq 1$, nous définissons la suite $\mathbf{r} = (r_n)_n$ par

$$r_n = n^{t(e_1+e_2)} \lambda(\mathbf{s})^{tn} n^u.$$

Alors

$$\mathbb{D}_{h,\varphi,\mathbf{r}}(\mathcal{S}'/\mathcal{S}) = \frac{b \cdot d(\mathcal{S},\mathcal{S}',\varphi)^t}{\lambda(\mathbf{s})^{t(e_1+e_2)}} \left(\ln \lambda(\mathbf{s})\right)^u.$$

(ii) Si $\lambda(\mathbf{s}) = 1$ et $e_1 + e_2 \neq 0$, nous définissons la suite $\mathbf{r} = (r_n)_n$ par

$$r_n = n^{t(e_1+e_2)} (\ln n)^u.$$

Alors

$$\mathbb{D}_{h,\varphi,\mathbf{r}}(\mathcal{S}'/\mathcal{S}) = \frac{b \cdot d(\mathcal{S},\mathcal{S}',\varphi)^t}{\lambda(\mathbf{s})^{t(e_1+e_2)}} \left(e_1+e_2\right)^u.$$

(iii) Si $\lambda(\mathbf{s}) = 1$ et $e_1 + e_2 = 0$, nous définissons la suite $\mathbf{r} = (r_n)_n$ par

$$r_n = 1.$$

Alors

$$\mathbb{D}_{h,\varphi,\mathbf{r}}(\mathcal{S}'/\mathcal{S}) = b \cdot d(\mathcal{S},\mathcal{S}',\varphi)^t \cdot \left(\ln d(\mathcal{S},\mathcal{S}',\varphi)\right)^u.$$

Nos résultats dans [5] sont plus précis puisque nous traitons la situation où la propriété de non dégénérescence des sommes n'est pas vérifiée. Cela conduit à de nombreux cas supplémentaires dont certains ne permettent pas d'élucider la limite. Cependant, la méthode reste la même. Il faut juste décrire plus précisément les asymptotiques et faire attention aux termes dominants qui s'annulent.

4.1.4 Application au calcul de taux d'entropie renormalisés

Afin d'illustrer la méthode générale, nous appliquons ici le théorème 10 à des taux d'entropie relative.

Entropie relative de Kullback-Leibler :

L'entropie relative de Kullback-Leibler est donnée par $\varphi(x,y) = x\ln(x/y)$ et $h(z) = z$. La fonction $\varphi(x,y)$ est de type quasi-poly-log avec $E_\varphi = \{(1,0;1,0),(1,0;0,1)\}$ et $a_{(1,0;1,0)} = -a_{(1,0;0,1)} = 1$. La fonction $h(z)$ est de type quasi-poly-log sur $\mathbb{R}\cup\{\pm\infty\}$ avec $b=1$, $t=1$ et $u=0$. Pour $(\mathbf{s},\mathbf{e})\in E_\varphi$, nous avons toujours

$$\lambda(\mathbf{s}) = \lambda(1,0) = 1, \quad c(\mathbf{s}) = 1 \quad \text{et} \quad e_1+e_2 = 1.$$

Les valeurs de $\lambda(\mathbf{s}) = \lambda(1,0)$ et $c(\mathbf{s}) = c(1,0)$ découlent du fait que la somme $\Lambda_n(\mathcal{S},\mathcal{S}';1,0;0,0) = 1$ pour tout entier n. Nous en déduisons que $E_\varphi^+ = E_\varphi$. Comme $\lambda(\mathbf{s}) = 1$ et $e_1+e_2 = 1 \neq 0$, nous sommes dans le cas (ii) du théorème 10. En posant $r_n = n$, nous obtenons que le taux d'entropie relative de Kullback-Leibler vérifie

$$\mathbb{D}_{z,x\ln(x/y),n}(\mathcal{S}'/\mathcal{S}) = d(\mathcal{S},\mathcal{S}',\varphi) = \lambda'_{s_1}(1,0) - \lambda'_{s_2}(1,0).$$

La propriété de non-dégénérescence dit simplement que $\lambda'_{s_1}(1,0) - \lambda'_{s_2}(1,0)$ est non nul.

Entropie relative de Rényi :

L'entropie relative de Rényi est donnée par $\varphi(x,y) = x^s y^{1-s}$ et $h(z) = (1-s)^{-1}\ln z$. La fonction $\varphi(x,y)$ est de type quasi-poly-log avec $E_\varphi = E_\varphi^+ = \{(s,1-s;0,0)\}$ et $a_{(s,1-s;0,0)} = 1$. La fonction $h(z)$ est de type quasi-poly-log sur $\mathbb{R}\cup\{\pm\infty\}$ avec $b = (1-s)^{-1}$, $t=0$ et $u=1$. Pour $(\mathbf{s},\mathbf{e})\in E_\varphi^+$, nous avons toujours $e_1+e_2 = 0$ mais il n'est pas possible de prévoir la valeur de $\lambda(\mathbf{s}) = \lambda(s,1-s)$. Nous savons que $\lambda(1,0) = \lambda(0,1) = 1$ mais ce sont les seules valeurs connues. Comme $e_1+e_2 = 0$, nous sommes soit dans le cas (i) avec $\lambda(s,1-s)\neq 1$, soit dans le cas (iii) si $\lambda(s,1-s) = 1$.

Si $\lambda(s,1-s)\neq 1$, le cas (i) du théorème 10 s'applique. La suite r_n est donnée par $r_n = n$ et le taux d'entropie relative vérifie

$$\mathbb{D}_{(1-s)^{-1}\ln z, x^s y^{1-s}, n}(\mathcal{S}'/\mathcal{S}) = \frac{1}{1-s}\ln\lambda(s,1-s).$$

Notez que lorsque $s\to 1$, le taux d'entropie relative de Rényi converge vers la divergence de Kullback-Leibler.

Si $\lambda(s,1-s) = 1$, le cas (iii) du théorème 10 s'applique. La suite r_n est donnée par $r_n = 1$ et le taux d'entropie relative vérifie

$$\mathbb{D}_{(1-s)^{-1}\ln z, x^s y^{1-s}, n}(\mathcal{S}'/\mathcal{S}) = \frac{1}{1-s}\ln c(s,1-s).$$

Comme la fonction $s\mapsto\lambda(s,1-s)$ est analytique, l'égalité $\lambda(s,1-s) = 1$ n'existe qu'en des points isolés de l'axe réel. Ce cas est mathématiquement possible mais en pratique, il correspond à un cas « rare ».

Entropie relative de Tsallis-Harvda-Charvat :

L'entropie relative de Tsallis-Harvda-Charvat est donnée par $\varphi(x,y) = x^s y^{1-s}$ et $h(z) = (1-s)^{-1}(z-1)$. La fonction $\varphi(x,y)$ est de type quasi-poly-log avec $E_\varphi = E_\varphi^+ = \{(s, 1-s; 0, 0)\}$ et $a_{(s,1-s;0,0)} = 1$. La fonction $h(z)$ est de type quasi-poly-log mais avec des équivalences différentes selon que $z_0 = 0$, $z_0 = \pm\infty$ ou $z_0 \neq 0$ constant. Précisément, nous avons

$$h(z) \underset{z\to\pm\infty}{\sim} \frac{1}{1-s} z, \quad h(z) \underset{z\to 0}{\sim} \frac{1}{s-1}, \quad h(z) \underset{\substack{z \to z_0 \\ z_0 \neq 0,1}}{\sim} \frac{1}{1-s}(z_0 - 1).$$

Trois cas sont possibles.

(*a*) si $\lambda(s, 1-s) < 1$, alors $z_0 = 0$ et le cas (i) du théorème s'applique avec $t = u = 0$, $b = (1-s)^{-1}$, $d = c(s, 1-s)$ et $e_1 + e_2 = 0$. La suite r_n vérifie $r_n = 1$ et

$$\mathbb{D}_{(1-s)^{-1}(z-1), x^s y^{1-s}, 1}(\mathcal{S}'/\mathcal{S}) = \frac{1}{1-s}.$$

(*b*) si $\lambda(s, 1-s) > 1$, alors $z_0 = \pm\infty$ et le cas (i) du théorème s'applique avec $t = 1$, $u = 0$, $b = (1-s)^{-1}$, $d = c(s, 1-s)$ et $e_1 + e_2 = 0$. La suite r_n vérifie $r_n = \lambda(s, 1-s)^n$ et

$$\mathbb{D}_{(1-s)^{-1}(z-1), x^s y^{1-s}, \lambda(s,1-s)^n}(\mathcal{S}'/\mathcal{S}) = \frac{1}{1-s} c(s, 1-s).$$

(*c*) si $\lambda(s, 1-s) = 1$, alors z_0 est constant et le cas (iii) du théorème s'applique avec $t = 0$, $u = 0$, $b = (1-s)^{-1}(z_0 - 1)$, $d = c(s, 1-s)$ et $e_1 + e_2 = 0$. La suite r_n vérifie $r_n = 1$ et

$$\mathbb{D}_{(1-s)^{-1}(z-1), x^s y^{1-s}, 1}(\mathcal{S}'/\mathcal{S}) = \frac{1}{1-s}(z_0 - 1).$$

Comme précédemment, la fonction $s \mapsto \lambda(s, 1-s)$ est analytique et l'égalité $\lambda(s, 1-s) = 1$ n'existe qu'en des points isolés de l'axe réel. Ce cas est mathématiquement possible mais en pratique, il correspond à un cas « rare ».

Pour des exemples d'applications à d'autres entropies (relatives), nous renvoyons à [5].

4.2 Combinatoire des motifs cachés avec les sources dynamiques

Nous décrivons ici le travail réalisé en collaboration avec Manuel Lladser sur le nombre d'occurrences d'un motif caché dans un texte produit par une source dynamique [9]. Dans ce travail, nous n'étudions pas la source mais plutôt les mots produits par la source et leurs occurrences.

4.2.1 Contexte

Les statistiques d'occurrences de motifs dans un texte sont au cœur de nombreuses applications en biologie moléculaire, fouille de texte, analyse de discours, sécurité, etc. Il existe plusieurs familles de motifs et pour beaucoup d'entre elles, des analyses probabilistes ont été réalisées.

Dans toute la suite, $\mathcal{A}$ est l'alphabet et $\mathcal{A}^\star$ est l'ensemble des mots finis sur $\mathcal{A}$. Une occurrence du mot w dans un texte T est une décomposition de T sous la forme $T = v_0 w v_1$ avec v_0 et v_1 dans $\mathcal{A}^\star$. Dans [69, 70, 68], Guibas et Odlyzko ont mis en évidence le rôle de l'autocorrélation dans le problème de trouver et compter les occurrences d'un mot (exact string matching problem) dans un texte produit par une source sans mémoire. Dans [111], Régnier et Szpankowski ont étendu les résultats et ont montré que le nombre d'occurrences exactes d'un mot dans un texte suit une loi

limite gaussienne lorsque le texte est produit par une chaîne de Markov. La recherche approchée d'un mot w correspond à la recherche exacte d'un mot parmi un ensemble de mots proches de w (selon une certaine distance). L'ensemble des mots *proches* est appelé le motif. Dans [110], Régnier et Szpankowski ont montré que le nombre d'occurrences lors d'une recherche approchée est asymptotiquement gaussien lorsque le texte est produit par une source sans mémoire.

La recherche exacte et approchée sont deux cas particulier de recherche des mots d'un langage régulier (ensemble de mots décrit par une expression régulière). Dans [105], Nicodème, Flajolet et Salvy mélangent la combinatoire analytique avec des outils classiques de théorie des langages et montrent que le nombre d'occurrences de mots d'un langage régulier suit asymptotiquement une loi limite gaussienne pour des sources sans mémoire ou des chaînes de Markov. Plus tard, Bourdon et Vallée étendent les résultats aux sources dynamiques [33]. Un motif caché un est k-uplet de mots $\mathcal{W} = (w_1, \ldots, w_k)$ avec $w_i \in \mathcal{A}^\star$. Une occurrence de $\mathcal{W}$ dans un texte T est une décomposition de T sous la forme $T = v_0 w_1 v_1 w_2 v_2 \ldots v_{k-1} w_k v_k$ où les v_i sont des mots de $\mathcal{A}^\star$. Les motifs généralisés ressemblent aux motifs cachés excepté que chaque mot w_i est remplacé par un langage régulier $\mathcal{R}_i$. Notons que les motifs généralisés englobent toutes les définitions précédentes de motifs. Flajolet, Szpankowski et Vallée ont prouvé que le nombre de motifs cachés dans un texte produit par une source sans mémoire suit une loi limite gaussienne [61]. Très récemment, Féray a étendu le résultat aux chaînes de Markov en s'appuyant sur des graphes de dépendance pondérés [59]. Bourdon et Vallée ont calculé l'espérance et la variance du nombre d'occurrences d'un motif généralisé dans le cadre des sources dynamiques et ils ont conjecturé l'existence d'une loi limite gaussienne lorsque le texte est produit par une source dynamique.

Dans cette section, nous étudions la loi limite du nombre d'occurrences d'un motif caché dans un texte produit par une source dynamique. La méthode utilisée dans [61] se base sur la convergence des moments et la combinatoire analytique mais elle ne s'adapte pas au contexte des sources dynamiques du fait de leurs corrélations. Afin de comprendre les difficultés sous-jacentes, nous avons décidé d'étudier le cas le plus simple de motif caché qui est composé de deux mots de une lettre. L'intérêt d'analyser ce *problème jouet* réside surtout dans la méthode mise en œuvre pour le résoudre.

4.2.2 Notations et résultats

Soit a, b deux lettres différentes de l'alphabet $\mathcal{A}$ et considérons le motif caché $\mathcal{M} = (a, b)$. Pour un mot $\mathbf{w} \in \mathcal{A}^\star$, nous posons

$$\begin{array}{lcl} ||\mathbf{w}|| & = & \text{Longueur du mot } \mathbf{w}, \\ |\mathbf{w}|_{(a,b)} & = & \text{Nombre d'occurrences du motif caché } (a,b) \text{ dans } \mathbf{w}, \\ |\mathbf{w}|_\ell & = & \text{Nombre d'occurrences de la lettre } \ell \text{ dans } \mathbf{w}, \\ C(\mathbf{w}) & = & \frac{|\mathbf{w}|_{(a,b)}}{|\mathbf{w}|_a} \text{ si } |\mathbf{w}|_a \neq 0, \text{ 0 sinon.} \end{array}$$

Par exemple pour $\mathbf{w} = abcabc$, $||\mathbf{w}|| = 6$, $|\mathbf{w}|_{(a,b)} = 3$, $|\mathbf{w}|_a = |\mathbf{w}|_b = |\mathbf{w}|_c = 2$ et $C(\mathbf{w}) = 3/2$. Le paramètre C est appelé le nombre d'occurrences normalisé de (a, b). Nous ne sommes pas parvenus à analyser le nombre d'occurrences du motif (a, b) mais seulement le nombre d'occurrences normalisés. Nos résultats dépendent de l'opérateur de transfert $\mathbf{K}(z, u)$

$$\mathbf{K}(z,u) = \sum_{\mathbf{w} \in a \cdot \mathcal{B}^\star} z^{||\mathbf{w}||} u^{|\mathbf{w}|_b} \mathbf{G}_{[\mathbf{w}]},$$

avec $\mathcal{B} = \mathcal{A} \backslash \{a\}$. La variable z porte la longueur des mots alors que la variable u marque les occurrences de la lettre b. En notant $\mathcal{C} = \mathcal{A} \backslash \{a, b\}$, les dictionnaires de l'analyse dynamique

conduisent à une expression explicite de $\mathbf{K}(z,u)$ donnée par

$$\mathbf{K}(z,u) = \Big(\mathbf{1} - z\left(\mathbf{G}_{[\mathcal{C}]} + u\mathbf{G}_{[b]}\right)\Big)^{-1} \circ (z\mathbf{G}_{[a]}).$$

Sur un espace fonctionnel adéquat, $\mathbf{K}(z,u)$ vérifie une propriété de Perron-Frobenius (voir la propriété 1). Il admet une unique valeur propre dominante $\lambda(z,u)$ et nous posons $A(z,u)$ la fonction bivariée

$$A(z,u) = \exp\left(\int_0^1 \log \lambda(z,u^t)\,dt\right).$$

et considérons la fonction $\sigma(u)$ telle que $\sigma(1) = 1$ et $A(\sigma(u),u) = 1$, pour tout u dans un voisinage de $u = 1$. Nous avons prouvé le théorème suivant.

Théorème 11 *Soit C le nombre d'occurrences normalisé du motif caché (a,b) dans un texte aléatoire de longueur n produit par une source dynamique holomorphe (voir [132]). La variable aléatoire C suit une loi limite gaussienne. Précisément, nous avons*

$$\lim_{n\to\infty} \mathbb{P}_n\left[\frac{C - \mathbb{E}_n[C]}{\sqrt{\mathbb{V}_n(C)}} \le x\right] = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{-t^2/2}\,dt,$$

où l'espérance et la variance de C satisfont

$$\begin{aligned}\mathbb{E}_n[C] &\sim -\sigma'(1)\cdot n,\\ \mathbb{V}_n[C] &\sim \Big(\sigma'(1)^2 - \sigma'(1) - \sigma''(1)\Big)\cdot n.\end{aligned}$$

La preuve du résultat repose sur la combinatoire analytique et les dictionnaires des systèmes dynamiques. Les textes qui contiennent exactement m occurrences de la lettre a appartiennent à l'ensemble $\mathcal{H}_m = \mathcal{B}^\star(a\mathcal{B}^\star)^m$. Pour un mot $\mathbf{w} = v_0 a v_1 a v_2 a v_3 \dots v_{m-1} a v_m$ de $\mathcal{H}_m$, le nombre d'occurrences normalisé du motif caché (a,b) dans $\mathbf{w}$ vérifie

$$C(\mathbf{w}) = \sum_{i=1}^{m} \frac{i}{m}\cdot |v_i|_b.$$

C'est un paramètre additif et le dictionnaire sur les opérateurs s'applique. L'opérateur de transfert pondéré associé à l'ensemble de mots $\mathcal{H}_m$ est alors

$$\mathbf{H}_m(z,u) = \left\{\prod_{i=m}^{1} \mathbf{K}(z,u^{i/m})\right\} \circ (\mathbf{1} - z\,\mathbf{G}_{[\mathcal{B}]})^{-1}.$$

Le produit signifie ici la composition des opérateurs. Il suffit de sommer sur m pour obtenir l'opérateur associé à $\mathcal{A}^\star$. Si la source est sans mémoire, le produit est un *vrai produit* de séries génératrices et les termes commutent. Il est alors possible d'utiliser la technique $\exp - \log$ combinée avec la formule d'Euler-Maclaurin pour obtenir finalement

$$\mathbf{H}_m(z,u) = \left(\exp\int_0^1 \log \mathbf{K}(z,u^t)\,dt\right)^m + \text{termes d'erreur}.$$

Pour une source avec mémoire, cette opération n'est plus possible. Une large part de la preuve consiste à montrer que l'on peut remplacer l'opérateur $\mathbf{K}(z,u)$ par sa partie spectrale dominante $\lambda(z,u)\mathbf{P}(z,u)$, avec $\mathbf{P}(z,u)$ le projecteur sur l'espace propre associé à $\lambda(z,u)$. En appliquant la

stratégie exp − log et la formule d'Euler-Maclaurin, la fonction $A(z,u)$ apparait naturellement et nous avons

$$\mathbf{H}_m(z,u) = A(z,u)^m \cdot \underline{\mathbf{P}}(z,u) + \text{termes d'erreur},$$

avec $\underline{\mathbf{P}}(z,u)$ un opérateur borné. En sommant sur m, le facteur $1/(1-A(z,u))$ intervient et admet un pôle simple en $z=\sigma(u)$. L'extraction des coefficients se fait de manière *assez classique* en tenant compte des erreurs apportées par la formule d'Euler Maclaurin sur les opérateurs. Ces erreurs nous empêchent d'utiliser le théorème des quasi-puissances de Hwang [79]. A la place, nous montrons la convergence ponctuelle de la transformée de Laplace du paramètre C vers la transformée de Laplace d'une loi gaussienne, ce qui permet de conclure.

4.3 Conclusion

Dans ce chapitre, nous avons étudié deux aspects des sources : le calcul explicite des entropies généralisées et la combinatoire d'un motif caché particulier dans un texte produit par une source dynamique.

Dans la première partie, nous avons étendu la notion usuelle de taux d'entropie en considérant une renormalisation adaptée aux fonctions d'entropie et aux sources considérées. Nous avons calculé explicitement les (h,φ)-taux d'entropie renormalisés pour une large classe de fonctions (h,φ) dites de type quasi-poly-log et pour des sources qui satisfont la propriété de quasi-puissance locale. La classe de fonctions (h,φ) de type quasi-poly-log est suffisamment large pour inclure les entropies (relatives) classiques de Shannon, Rényi, Kullback-Leibler, Sharma-Mittal, etc. De son côté, la propriété de quasi-puissance locale s'applique à toutes les sources classiques sous des hypothèses très simples à vérifier : sources sans-mémoire, chaînes de Markov, sources dynamiques, etc.

Il est probablement possible d'étendre notre méthode à des entropies qui ne sont pas quasi-poly-log mais qui sont de la forme

$$h\Big(\sum_{\mathbf{w}\in\mathcal{A}^n} \varphi_1(p_{\mathbf{w}}, p'_{\mathbf{w}}) \odot \sum_{\mathbf{w}\in\mathcal{A}^n} \varphi_2(p_{\mathbf{w}}, p'_{\mathbf{w}})\Big) \quad \text{avec} \quad \odot \in \{+,-,\times,/\}$$

et φ_1, φ_2 deux fonctions quasi-poly-log (voir [58] pour des exemples). En considérant des fonctions quasi-poly-log $\varphi(x,y,z)$ avec trois variables, une autre extension possible serait les entropies relatives de Bregman [126].

Toutefois, il existe des entropies pour lesquelles notre approche ne s'applique pas. Les entropies relatives dont $\varphi(x,y)$ contient des termes de la forme $(x^\beta \pm y^\beta)^\alpha$ avec α et β non entiers ne sont pas quasi-poly-log. C'est le cas des entropies relatives de type 5 dans le tableau 4.2. Les entropies trigonométriques introduites dans [116] font intervenir les fonctions sinus et cosinus (hyperboliques ou pas). Notre approche ne s'adapte pas non plus à ce type d'entropies.

Dans la deuxième partie, nous avons montré que le nombre d'occurrences du motif caché (a,b) normalisé suit une loi limite gaussienne. Outre le résultat, l'apport essentiel de ce travail est double. D'une part, nous pouvons maintenant envisager d'étudier des motifs plus généraux. Ensuite, nous avons démontré une formule de type Euler-Maclaurin pour les opérateurs de transfert. Ces travaux sur les motifs cachés restent inachevés. Des échanges avec Manuel Lladser nous ont convaincu que l'on pouvait aller plus loin. Il est probablement possible d'analyser le cas de deux mots distincts en combinant une approche par automate et en marquant certains états. Pour le motif (a,b), nos analyses s'appuient sur la série génératrice bivariée avec une variable qui porte la taille du texte et une autre qui porte le nombre d'occurrences renormalisé. Il est

clair que la série trivariée avec une variable supplémentaire qui porte le nombre d'occurrences de a peut s'obtenir facilement. Cette série permettrait d'étudier la loi conjointe du nombre d'occurrences de a et du nombre d'occurrences renormalisé de (a, b). Nous en déduirions peut-être la loi du nombre d'occurrences de (a, b). Un autre axe de recherche serait aussi d'adapter les travaux de Féray [59] sur les chaînes de Markov et les graphes de dépendances au cas des sources dynamiques.

Chapitre 5

Hypergraphes et fouille de données

Ce chapitre décrit les principaux résultats de l'article suivant :
– [4] **An average study of hypergraphs and their minimal transversals.**
J. David, L. Lhote, A. Mary, F. Rioult.
Theoretical Computer Science, vol. 596, pages 124–141, 2015.

L'objet d'étude dans ce chapitre est l'hypergraphe et ses traverses minimales. Bien que la structure d'hypergraphe est très différente des algorithmes euclidiens ou des sources, la philosophie de ce chapitre reste la même que celles des chapitres 2 à 4 : nous souhaitons comprendre le comportement probabiliste des traverses minimales dans un hypergraphe aléatoire. Nous obtenons plusieurs résultats. Nous estimons le nombre moyen de traverses minimales dans un hypergraphe aléatoire sous des modèles qui généralisent celui de Erdős-Rényi pour les graphes [56, 57]. Nous nous intéressons aussi à la complexité générique [81] de l'algorithme MTMiner qui énumère les traverses minimales selon une stratégie inspirée de la fouille de données.

D'un point de vue plus personnel, ce travail s'inscrit dans la lignée d'autres travaux à l'interface entre la fouille de données et l'Analyse d'algorithmes [16, 17, 13]. J'ai commencé à m'intéresser aux problématiques de la fouille de données autour de 2004 pendant ma thèse. Des discussions informelles avec des doctorants du laboratoire m'ont convaincu de l'intérêt de réaliser des analyses probabilistes dans le cadre de l'extraction de motifs dans les bases de données (Pattern Mining). L'extraction de motifs est un domaine qui vise à extraire de manière automatique de l'information des bases sous forme de motifs. La nature des motifs dépend du contexte applicatif et des objectifs de l'extraction. Il existe de nombreux types de motifs (fréquents, fermés, libres, émergents, séquentiels, bordure négative, etc.) et de nombreux algorithmes dédiés ou génériques issus du Pattern Mining. Les algorithmes s'appuient sur des structures de données bien connues de la communauté *analyses d'algorithmes* comme les tries, les séquences, les graphes, les hypergraphes, etc. Cependant à notre connaissance, il existe très peu d'interactions entre les deux communautés.

L'analyse d'algorithmes et le Pattern Mining ont beaucoup à gagner en collaborant. Dans la plupart des cas, la fouille de données ne dispose que des bornes dans le pire des cas qui sont presque toujours exponentielles. En pratique, les algorithmes sont plus ou moins efficaces selon le réglage des paramètres et les bornes dans le pire des cas ne suffisent pas à expliquer ces phénomènes. Un des objectifs des analyses probabilistes serait de compter le nombre de motifs *attendus* dans une base de données et d'expliquer ainsi certains phénomènes. Autrement dit avec les analyses, on peut espérer quantifier *a priori* l'information à extraire dans une base de données sans avoir besoin de lancer les algorithmes d'extraction. Une étude systématique des algorithmes permettrait aussi de les comparer afin de déterminer lequel est le meilleur dans un

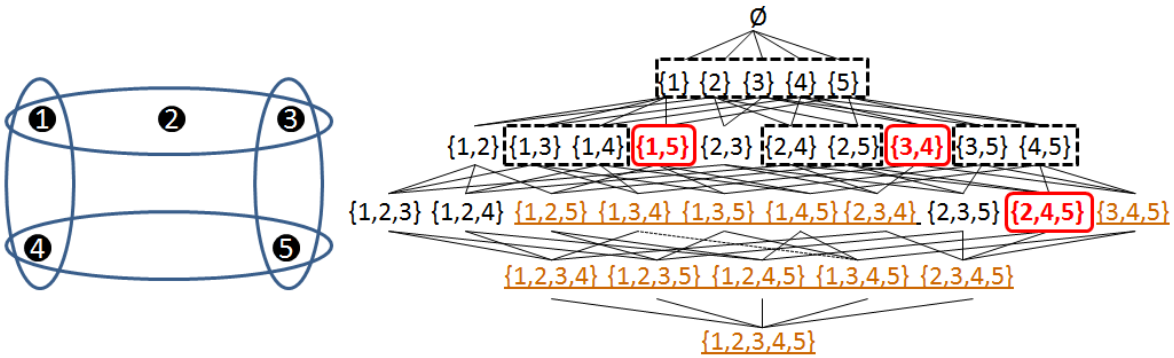


FIGURE 5.1 – A gauche : un hypergraphe $\mathcal{H} = (V, \mathcal{E})$ où $V = \{1, 2, 3, 4, 5\}$ et $\mathcal{E} = \{\{1, 2, 3\}, \{1, 4\}, \{3, 5\}, \{4, 5\}\}$. A droite : le treillis des motifs sur V. Les traverses minimales sont encadrées avec un trait plein. Les traverses non-minimales sont soulignées. Les non-redondants qui ne sont pas des traverses sont encadrés en pointillés.

contexte donné. Pour que les analyses aient un intérêt en fouille de données, les modèles probabilistes doivent être réalistes. Cette contrainte imposera probablement à l'analyse d'algorithmes d'étendre ses techniques (comme la combinatoire analytique) afin de tenir compte de modèles aléatoires plus complexes. Ces nouvelles techniques pourront ensuite être utilisées dans d'autres contextes.

Cette approche a motivé le dépôt en 2015 du PEPS HYDrATA, de l'appel à projet FASCIDO sur la science des données, qui a été accepté puis renouvelé en 2016 et dont je suis porteur.

Plan. La section 5.1 introduit les notions principales autour des hypergraphes et présente la problématique générale de l'énumération des traverses minimales avec les résultats connus. La section 5.2 décrit un exemple de lien entre les traverses minimales et l'extraction des motifs et remet en perspective les travaux de ce chapitre avec mes précédents travaux. Les modèles aléatoires utilisés pour les analyses des traverses minimales sont décrits à la section 5.3. Les sections 5.4 et 5.5 présentent les résultats obtenus dans chacun des modèles. Dans la section 5.6, nous nous intéressons au comportement probabiliste de l'algorithme MTMINER. Finalement, nous concluons avec des perspectives d'amélioration des travaux (section 5.7).

5.1 Hypergraphes et Extraction de motifs

5.1.1 Hypergraphes et traverses minimales

Les hypergraphes généralisent la notion de graphe en autorisant une arête à connecter en même temps plusieurs sommets. Un hypergraphe est une paire $\mathcal{H} = (V, \mathcal{E})$ où $V = \{1, 2, \ldots, n\}$ est l'ensemble des sommets et $\mathcal{E} = \{E_1, \ldots, E_m\}$ est l'ensemble des hyperarêtes avec $E_i \subseteq V$ pour tout i. Par exemple, l'hypergraphe $\mathcal{H} = (V, \mathcal{E})$ avec $V = \{1, 2, 3, 4, 5\}$ et $\mathcal{E} = \{\{1, 2, 3\}, \{1, 4\}, \{3, 5\}, \{4, 5\}\}$ est représenté à la figure 5.1 (à gauche).

Une *traverse* est un ensemble de sommets qui intersecte toutes les hyperarêtes. En reprenant l'exemple précédent, nous avons $\{1, 5\}$ ou $\{1, 3, 4\}$ qui sont des traverses de $\mathcal{H}$ alors que $\{1, 2\}$ ou $\{1, 3\}$ ne le sont pas. La figure 5.1 représente sous forme de treillis (pour l'inclusion), tous les ensembles de sommets possibles. Les ensembles soulignés ou encadrés avec des traits pleins

sont des traverses.

Un ensemble de sommets X est dit non-redondant si dès qu'on lui retire un sommet i, $X\backslash\{i\}$ intersecte strictement moins d'hyperarêtes que X. Une autre définition équivalente est que pour tout sommet i de X, il existe une hyperarête $E_j \in \mathcal{E}$ telle que $E_j \cap X = \{i\}$. En reprenant notre exemple, $\{1,3\}$ ou $\{1,5\}$ sont non-redondants contrairement à l'ensemble $\{2,3,5\}$ où 2 peut être enlevé sans changer le nombre d'hyperarêtes intersectées. Dans le treillis de la figure 5.1, les ensembles encadrés (avec des pointillés ou des traits pleins) sont des ensembles non-redondants.

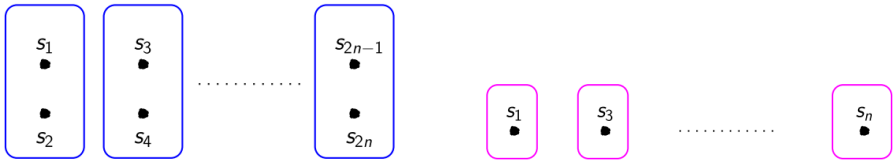
Pour terminer, une traverse $X \subset V$ est dite *minimale* si elle est non-redondante. Autrement dit, une traverse minimale est un ensemble de sommets qui intersecte toutes les hyperarêtes et tel que tout sous-ensemble strict n'est plus une traverse. Avec notre exemple, $\{1,5\}$ est une traverse minimale, $\{1,4\}$ est non-redondant mais n'est pas une traverse, $\{1,2,5\}$ est une traverse mais n'est pas non-redondant (2 peut être enlevé sans diminuer le nombre d'hyperarêtes intersectées) et $\{1,2,3\}$ n'est ni une traverse, ni non-redondant.

Pour un hypergraphe $\mathcal{H}$, l'ensemble de ses traverses minimales forme un autre hypergraphe appelé *hypergraphe transversal* de $\mathcal{H}$ et noté $\mathcal{T}(\mathcal{H})$. Avec notre exemple, nous avons $\mathcal{T}(\mathcal{H}) = \{\{1,5\},\{3,4\},\{2,4,5\}\}$.

5.1.2 Énumération des traverses minimales : contexte

Étant donné un hypergraphe $\mathcal{H}$, le problème THG (Transversal Hypergraph Generation) consiste à générer l'hypergraphe transversal $\mathcal{T}(\mathcal{H})$ de $\mathcal{H}$. Le problème THG apparait dans de nombreux domaines comme l'intelligence artificielle et la logique [52, 53, 54], la biologie [41], la fouille de données et l'apprentissage automatique [71], les télécommunications [117], *etc.* Pour une liste plus complète d'applications, nous renvoyons à [72]. Le problème de décision associé, appelé problème THD (Transversal Hypergraph Decision), consiste à déterminer si un hypergraphe $\mathcal{H}_2$ est l'hypergraphe transversal d'un premier hypergraphe $\mathcal{H}_1$.

Dans le pire des cas, un hypergraphe peut avoir un nombre exponentiel de traverses minimales en le nombre de sommets ou d'hyperarêtes. Par exemple, l'hypergraphe contenant $2n$ sommets et n hyperarêtes disjointes de cardinal 2 a exactement 2^n traverses minimales [4]. Dans le meilleur des cas, chaque sommet est une hyperarête et le nombre de traverses minimales est 1. La figure qui suit illustre ces cas extrémaux.



2^n trav. min.
$\leq \binom{2n}{n}$

1 trav. min

Comme la sortie du problème THG peut être exponentielle en la taille de l'entrée, le problème n'appartient pas à la classe des problèmes polynomiaux. Une question encore ouverte est de savoir si le problème est output-polynomial, autrement dit si la complexité dans le pire des cas est polynomiale en la taille de l'entrée et de la sortie. La complexité du problème de génération THG est étroitement liée à celle du problème de décision THD. Il est clair que si THG est output-polynomial dans le pire des cas, alors THD peut être résolu en temps polynomial. De plus, THD est clairement dans la classe co-NP et Eiter [52] a démontré que s'il était co-NP-complet, alors THG n'est pas output-polynomial à moins que P=co-NP.

4. Il faut choisir un sommet parmi 2 dans chaque hyperarête.

L'algorithme de Fredman et Khachiyan [62] est l'algorithme avec la meilleure borne de complexité connue pour générer les traverses minimales. Il est output-quasi-polynomial et son temps d'exécution est d'ordre $N^{O(\log N)}$ avec N la taille de l'entrée plus celle de la sortie. D'autres algorithmes sont en pratique plus efficaces (voir [55]) mais aucune borne de complexité n'est donnée. Dans ce chapitre, nous étudions l'algorithme MTMiner défini par Hébert, Bretto et Crémilleux [75] qui s'inspire des algorithmes de type Apriori en fouille de données [21]. Il est clairement output-exponentiel dans le pire des cas mais nous verrons que sa complexité générique [81] peut être bien différente.

Contrairement à tous les résultats précédents, nous adoptons dans ce chapitre une approche probabiliste. Il existe plusieurs analyses probabilistes de la structure d'hypergraphe. Un modèle très répandu est la distribution uniforme sur les hypergraphes k-uniformes dont toutes les hyperarêtes sont de cardinal k [20, 50, 94]. Ravelomanana a étudié dans [109] la création et la croissance des composantes dans des processus continus d'hypergraphes aléatoires. Plus récemment, De Panafieu [44] a développé un modèle aléatoire d'hypergraphes non-uniformes et a étudié leurs structures lors de la naissance de composantes complexes. Il existe une analyse probabiliste des traverses minimales d'hypergraphes aléatoires. Dans [125], les auteurs montrent qu'avec la distribution uniforme sur les hypergraphes à n sommets (sans contrainte sur le nombre d'hyperarêtes), le problème THG est presque sûrement output-polynomial. En fait avec ce modèle, les hypergraphes à n sommets ont presque sûrement un nombre exponentiel d'hyperarêtes et même l'algorithme naïf qui parcourt tout l'espace de recherche est presque sûrement output-polynomial. A notre connaissance, ce sont les seuls résultats probabilistes connus sur les traverses minimales.

5.2 Liens avec la fouille de données

En collaboration avec Arnaud Soulet et François Rioult, je me suis intéressé pendant ma thèse à l'analyse probabiliste des motifs fréquents et des motifs fermés dans une base de données *tabulaire* aléatoire [16, 17]. Mes travaux sur les traverses minimales découlent de ces premiers travaux. Dans ce contexte, une base de données est un triplet $\mathcal{B} = (O, A, \mathcal{R})$ avec $\mathcal{R}$ une relation binaire entre deux ensembles finis $O = \{o_1, o_2, \ldots, o_m\}$ et $A = \{a_1, a_2, \ldots, a_n\}$. Les éléments de A sont appelés *attributs* (ou *items*) alors que les éléments de O sont les *objets* (ou *transactions*). En pratique, les attributs sont des propriétés que les objets possèdent ou non. Par exemple, l'objet *voiture* peut avoir comme attributs *couleur blanche, citroen, occasion*.

La base de données peut se représenter de manière transactionnelle ou bien avec une matrice. Dans la représentation transactionnelle, chaque ligne décrit un objet et contient la liste des attributs en relation avec cet objet. Dans la représentation matricielle, chaque ligne de la matrice représente un objet et chaque colonne représente un attribut. Le coefficient d'indice (i, j) vaut 1 si et seulement l'objet o_i est en relation avec l'attribut a_j. La figure suivante donne les représentations matricielle et transactionnelle de la même base de données.

	❶	❷	❸	❹	❺
o_1	0	0	0	1	1
o_2	0	1	1	0	1
o_3	1	1	0	1	0
o_4	1	1	1	0	0

o_1 : 4, 5
o_2 : 2, 3, 5
o_3 : 1, 2, 4
o_4 : 1, 2, 3

Sur le site du FIMI [5] (Frequent Itemset Mining Implementations), il est possible de télécharger de nombreuses données réelles mises sous forme transactionnelle.

Nous appelons motif tout ensemble d'attributs. Pour un seuil de fréquence $\gamma \geq 0$, un motif est dit γ-fréquent s'il est présent dans au moins γ objets. Par exemple, le motif $X = \{2, 3\}$ est présent dans deux objets o_2 et o_4. Autrement dit, $X \cap o_2 = X \cap o_4 = X$. Le motif X est par définition 1 ou 2-fréquent mais n'est ni 3 ni 4-fréquent. Voici la liste des motifs 1-fréquents, ordonnés selon leur taille, de l'exemple précédent :

$$\emptyset,$$

$$\{1\},\ \{2\},\ \{3\},\ \{4\},\ \{5\},$$

$$\{1,2\},\ \{1,3\},\ \{1,4\},\ \{2,3\},\ \{2,4\},\ \{2,5\},\ \{3,5\},\ \{4,5\},$$

$$\{1,2,3\},\ \{1,2,4\},\ \{2,3,5\}.$$

Pendant ma thèse, j'ai réalisé avec mes co-auteurs dans [16, 17] les premières analyses probabilistes du nombre de motifs fréquents dans une base de données aléatoire. Nous avons considéré plusieurs modèles où les lignes de la représentation matricielle sont générées par des sources sur l'alphabet $\{0, 1\}$. Les sources étudiées étaient par exemple des sources de Bernoulli de paramètre fixé, des chaînes de Markov et même des sources dynamiques dans [13]. Dans le pire des cas (matrice avec uniquement des 1), le nombre de motifs fréquents est exponentiel en le nombre d'attributs. A l'inverse avec une matrice nulle, seul le motif vide est fréquent. Nos analyses donnent des informations plus fines. Lorsque le seuil de fréquence γ croît linéairement avec le nombre d'objets, le nombre de motifs fréquents est en moyenne polynomial en le nombre d'attributs. Si au contraire le seuil reste constant alors que la base grandit, le nombre de motifs fréquents devient exponentiel en le nombre d'attributs et polynomial en le nombre d'objets. Ces résultats ont été obtenus avec des modèles assez généraux. Dans [15], les expériences sur les bases du FIMI faites avec un modèle de Bernoulli multivarié montrent que les résultats théoriques sont très proches de la réalité.

Fort de ces résultats positifs, nous nous sommes intéressés aux motifs de la bordure négative. Un motif appartient à la bordure négative s'il n'est pas γ-fréquent et si tous ses sous-ensembles stricts le sont. Pour simplifier, prenons $\gamma = 1$. Alors les motifs de la bordure négative avec l'exemple précédent sont :

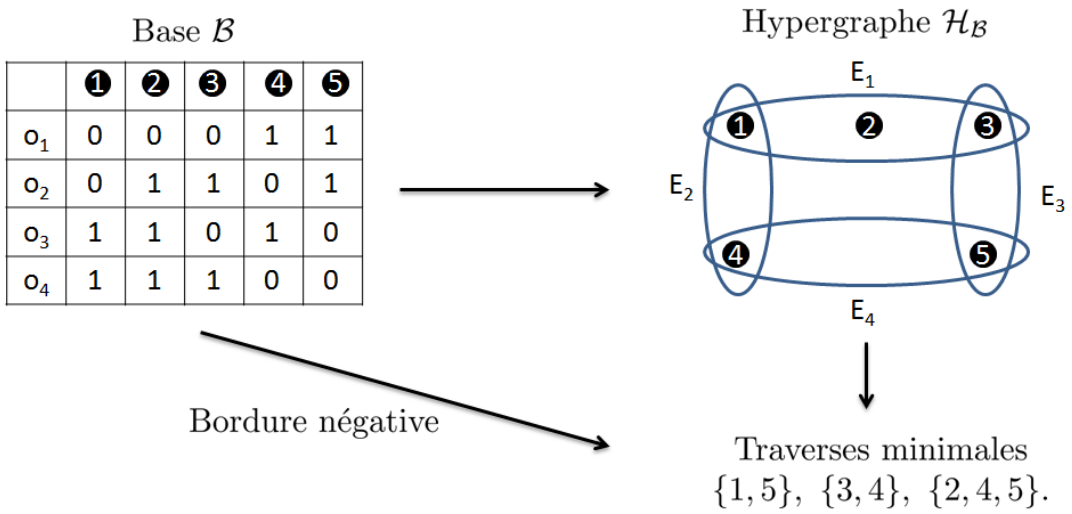
$$\{1,5\},\ \{3,4\},\ \{2,4,5\}.$$

Pour les trouver, il suffit de vérifier la propriété avec les motifs qui manquent parmi les motifs fréquents. D'un point de vue algorithmique, calculer les motifs de la bordure négative est équivalent à générer les traverses minimales d'un hypergraphe. Précisément si $\mathcal{B} = (O, A, \mathcal{R})$, alors l'hypergraphe $\mathcal{H}_\mathcal{B} = (V, \mathcal{E})$ est défini par $V = \{1, 2, \ldots, n\}$ et $\mathcal{E} = \{e_1, e_2, \ldots, e_m\}$ avec pour $i = 1..m$,

$$e_i = \{j \in \{1, 2, \ldots, n\} \mid \text{l'objet } o_i \text{ n'est pas en relation avec l'attribut } a_j\}.$$

Les traverses minimales de $\mathcal{H}_\mathcal{B}$ sont exactement les motifs de la bordure négative de $\mathcal{B}$. La figure suivante illustre ce résultat avec l'exemple de base de données précédent. Il se trouve que l'hypergraphe $\mathcal{H}_\mathcal{B}$ est donné par l'hypergraphe de la section précédente dont les traverses minimales sont bien les motifs de la bordure négative calculée un peu plus tôt.

5. `http://fimi.ua.ac.be/`



	❶	❷	❸	❹	❺
o_1	0	0	0	1	1
o_2	0	1	1	0	1
o_3	1	1	0	1	0
o_4	1	1	1	0	0

L'analyse probabiliste des motifs de la bordure négative revient à faire l'analyse probabiliste des traverses minimales. Comme les modèles de Bernoulli donnaient de bons résultats avec les motifs fréquents, nous avons décidé de conserver ces modèles pour l'analyse des traverses minimales.

5.3 Modèles d'hypergraphes aléatoires

Nous considérons deux modèles probabilistes d'hypergraphes aléatoires. Le premier est paramétré par un seul paramètre $p \in]0,1[$ alors que le second a autant de paramètres que de sommets. Dans les deux cas, les modèles partagent les hypothèses suivantes.

(C_1) Le nombre d'hyperarêtes est au plus polynomial en le nombre de sommets n : il existe $(\alpha, \beta) \in \mathbb{R}^+$ tel que $m = \beta n^\alpha$.

(C_2) Si $\mathcal{H} = (V, \mathcal{E})$ avec $V = \{1, 2, \ldots, n\}$ et $\mathcal{E} = \{E_1, E_2, \ldots, E_m\}$, on note $X_{i,j}$ la variable aléatoire qui vaut 1 si l'hyperarête E_i contient le sommet j, 0 sinon. Alors la famille $(X_{i,j})_{i=1..m, j=1..n}$ forme une famille indépendante de variables aléatoires.

Dans les applications, la condition (C_1) est raisonnable car on trouve assez rarement des bases dont β dépasse 6 ou 7. Pour $\beta = 6$ et un nombre de sommets $n = 100$, le nombre d'hyperarêtes m seraient de l'ordre du milliard ! D'un point de vue théorique, nos résultats n'exigent pas tous la condition (C_1) et peuvent être étendus au cas où m est exponentiel en n. Dans ce cas, la génération des traverses minimales devient trivialement output-polynomial puisque la taille de l'entrée est exponentielle en n tout comme le nombre maximal de traverses minimales.

La condition (C_2) est discutable, surtout dans le cadre de la fouille de données. Cette condition suppose qu'il n'y a pas de corrélations entre les sommets alors que la fouille de données cherche justement à mettre en évidence ces corrélations. C'est contradictoire mais nos expériences avec les motifs fréquents montrent que ce type d'hypothèses n'est pas dénué d'intérêt. De plus, constater que des motifs ont un comportement qui s'écarte d'un comportement aléatoire est aussi une information.

5.3.1 Modèle à un paramètre

Le premier modèle est le plus simple et s'inspire du modèle $G(n, p)$ introduit par Erdős-Rényi [56, 57] pour les graphes aléatoires.

Définition 4 (Modèle HG(n, m, p).) *Soit $p \in]0,1[$ et $(m,n) \in \mathbb{N}^2$. Le modèle $HG(n,m,p)$ suppose que la famille de variables aléatoires $(X_{i,j})_{i=1..m,j=1..n}$ forme une famille indépendante et identiquement distribuée de variables aléatoires de même loi de Bernoulli de paramètre p.*

Le modèle à un paramètre entraine que tous les sommets ont le même comportement probabiliste. C'est une hypothèse non réaliste mais ce modèle simple permet des analyses probabilistes plus précises. De plus, nous nous appuierons sur les résultats dans ce premier modèle pour les étendre au modèle multivarié.

Le modèle à un paramètre suppose que la probabilité p est fixée. En fait, nos résultats s'appliquent lorsque p tend *lentement* vers 0 ou 1. Précisément, nous supposons que p vérifie les inégalités

$$1 - e^{-\frac{1}{\ln n}} < p < e^{-\frac{1}{\ln n}}.$$

Cet encadrement se justifie à l'aide des calculs mais sera très utile dans le modèle multivarié.

5.3.2 Modèle à plusieurs paramètres

Le gros défaut du premier modèle est qu'il donne le même comportement à tous les sommets. Dans le modèle multivarié, nous ne supposons plus que les sommets appartiennent à une hyperarête avec la même probabilité. Chaque sommet a sa propre probabilité ce qui est plus proche des bases de données réelles.

Définition 5 (Modèle HG(n, m, g)) *Soit $g : \mathbb{N} \mapsto]0,1[$ une fonction. Pour $i \in \mathbb{N}$, nous posons $p_i = g(i)$ et $q_i = 1 - p_i$.*

Le modèle $HG(n,m,g)$ suppose que la famille de variables aléatoires $(X_{i,j})_{i=1..m,j=1..n}$ forme une famille indépendante de variables aléatoires. De plus pour tout $i = 1..m$ et $j = 1..n$, $X_{i,j}$ suit une loi de Bernoulli de paramètre p_j.

Avec ce modèle, la fréquence d'apparition de chaque sommet peut être contrôlée. Les analyses mettent en évidence trois types de sommets qui ont un rôle différent dans les asymptotiques.

- Les sommets *ubiquitaires* sont présents dans presque toutes les hyperarêtes. Pour une constante $x > 0$, nous dirons que le sommet i est ubiquitaire si $p_i > 1 - \frac{x}{m}$. L'ensemble des sommets ubiquitaires est noté U.
- Les sommets *rares* sont présents dans peu d'hyperarêtes. La probabilité d'un sommet rare i vérifie $p_i < 1 - e^{-\frac{1}{\ln n}}$. Nous notons R l'ensemble des sommets rares.
- Les autres sommets forment l'ensemble $S = V \setminus \{U \cup R\}$.

Nous verrons que le nombre de sommets rares est un paramètre important pour contrôler le nombre moyen de traverses minimales.

5.4 Résultats obtenus dans le modèle univarié

Comme une traverse minimale est à la fois une traverse et non-redondante, nous avons étudié le comportement probabiliste des traverses, des non-redondants et finalement des traverses minimales. Cette section résume les résultats obtenus pour le modèle à un paramètre.

5.4.1 Nombre moyen de traverses

Soit X un ensemble de sommets de cardinal $j \in [1,n]$. Comme chaque sommet appartient à une hyperarête avec la probabilité p, la probabilité que X intersecte une hyperarête $E_i \in \mathcal{E}$ est

$1-q^j$ avec $q=1-p$. La probabilité que X intersecte les m hyperarêtes est alors donnée par $(1-q^j)^m$. La proposition suivante décrit la proportion de traverses par niveau (à cardinal fixé) et découle de calculs élémentaires qui consistent essentiellement à remplacer j par des valeurs bien choisies.

Proposition 9 *Pour $j=1..n$, T_j est la variable aléatoire qui compte le nombre de traverses de cardinal j dans un hypergraphe. Dans le modèle aléatoire $HG(n,m,p)$, le nombre moyen de traverses de taille $j\in[1,n]$ est*

$$\mathbb{E}[T_j]=\binom{n}{j}(1-q^j)^m. \tag{5.1}$$

Considérons j de la forme

$$j=\log_{\frac{1}{q}} m+\log_{\frac{1}{q}} x, \qquad x\in\left]\frac{1}{m},+\infty\right[.$$

(i) Si $x\leq 1/\left(\ln n\times\log_{\frac{1}{q}} m\right)$, le nombre moyen de traverses de cardinal j tend vers zéro, i.e., $\mathbb{E}[T_j]=o(1)$.
(ii) Si x tend vers 0 (avec x minoré par $1/m$ afin de rester positif), le nombre de traverses de cardinal j est négligeable par rapport au nombre de sous-ensembles de sommets de même taille, i.e. $\mathbb{E}[T_j]=o\left(\binom{n}{j}\right)$.
(iii) Si $x=\Theta(1)$, le nombre moyen de traverses est proportionnel au nombre d'ensembles de cardinal j. Précisément, $\mathbb{E}[T_j]\sim\binom{n}{j}\exp(-1/x)$.
(iv) Si x tend vers $+\infty$, alors presque tous les ensembles de taille j sont des traverses, i.e., $\mathbb{E}[T_j]\sim\binom{n}{j}\left(1-\frac{1}{x}\right)$.
(v) Si $x\geq p/\ln\ln n$, alors l'écart-type de T_j vérifie

$$\sigma[T_j]=\mathcal{O}\left(\mathbb{E}[T_j]\frac{\ln n}{\sqrt{n}}\right).$$

L'inégalité de Bienaymé-Chebyshev entraine que le nombre de traverses de cardinal j est presque sûrement proche de son espérance et vérifie pour tout $\epsilon>0$,

$$\mathbb{P}\left(\left|\frac{T_j}{\mathbb{E}[T_j]}-1\right|>\epsilon\right)=O\left(n^{-1}\epsilon^{-2}\ln^2 n\right)\to 0.$$

Notez que par rapport à l'article [4], nous avons corrigé la borne de x dans le cas (i) où une parenthèse est mal placée. La propriété de concentration autour de la moyenne décrite par le cas (v) est fondamentale pour obtenir des bornes inférieures presque sûres sur le nombre de traverses minimales. Cette propriété est aussi intéressante puisqu'elle s'applique aux cas (iii) et (iv) mais aussi en partie au cas (ii).

5.4.2 Nombre moyen de non-redondants

Un ensemble X de cardinal j est non-redondant si pour tout sommet $k\in X$, il existe une hyperarête E_i telle que $E_i\cap X=\{k\}$. La probabilité que $E_i\cap X=\{k\}$ est pq^{j-1}. La probabilité qu'il n'existe pas de sommet k de X tel que $E_i\cap X=\{k\}$ est $1-jpq^{j-1}$. En séparant les

hyperarêtes contenant exactement un sommet des autres hyperarêtes en contenant 0 ou plus de deux, nous trouvons que la probabilité que X soit non-redondant est

$$\mathbb{P}[X \text{ non redondant}] = \sum_{\substack{j \leq \ell \leq m \\ k_1 + \ldots + k_j = \ell \\ \forall i = 1..j,\ k_i \geq 1}} \binom{m}{k_1, \ldots, k_j} (pq^{j-1})^{\ell} (1 - jpq^{j-1})^{m-\ell}.$$

Il est possible de majorer la probabilité précédente par

$$\mathbb{P}[X \text{non redondant}] \leq \frac{m!}{(m-j)!} (pq^{j-1})^{j}. \tag{5.2}$$

Cette formule revient à choisir une hyperarête E_{i_k} pour chaque sommet k de X (il y a $m!/(m-j)!$ choix possibles). Ensuite X est non-redondant dès que $X \cap E_{i_k} = \{k\}$ pour tout k. La probabilité de cet évènement est $(pq^{j-1})^{j}$. La proposition suivante s'appuie sur l'inégalité 5.2 et montre que le nombre moyen de non-redondants est quasi-polynomial.

Proposition 10 *Le nombre moyen de non-redondants dans le modèle aléatoire $HG(n, m, p)$ est d'ordre*

$$\mathcal{O}\left(\binom{n}{j_0} \frac{m!}{(m-j_0)!} (pq^{j_0-1})^{j_0} \ln n\right),$$

avec

$$j_0 = \left\lceil \frac{1}{2} \log_{\frac{1}{q}} mn - \frac{1}{2} \log_{\frac{1}{q}} \log_{\frac{1}{q}} mn + \frac{1}{2} \log_{\frac{1}{q}} 2p \right\rceil,$$

où $\lceil x \rceil$ est le plus petit entier supérieur ou égal à x.

Il est possible de calculer une asymptotique précise en insérant la valeur de j_0 dans le O. En ne gardant que le terme *dominant*, le comportement quasi-polynomial devient plus clair puisque le terme dans le O est d'ordre

$$\exp\left(\frac{1}{4|\ln q|} (\ln mn)^2 (1 + o(1))\right).$$

Nous ne sommes pas parvenus à faire une analyse par niveau précise des non-redondants mais ce résultats suffit pour donner une borne de complexité de l'algorithme MTMiner.

5.4.3 Nombre moyen de traverses minimales

Le raisonnement sur la probabilité d'être non-redondant s'adapte aux traverses minimales. Un ensemble X de cardinal j est une traverse minimale s'il est non-redondant et s'il est une traverse. Pour toute hyperarête E_i, la probabilité que X intersecte E_i en exactement un sommet est pq^{j-1}. Comme X est non-redondant, tout sommet de X intersecte au moins une hyperarête en exactement 1 sommet. D'un autre côté, la probabilité que E_i intersecte X en au moins 2 sommets est $1 - q^j - jpq^{j-1}$. En séparant les hyperarêtes contenant exactement un sommet des hyperarêtes en contenant au moins 2, nous trouvons que la probabilité que X soit une traverse minimale est

$$\mathbb{P}[X \text{ est une traverse minimale }] = \sum_{\substack{j \leq \ell \leq m \\ k_1 + \ldots + k_j = \ell \\ \forall i = 1..j,\ k_i \geq 1}} \binom{m}{k_1, \ldots, k_j} (pq^{j-1})^{\ell} (1 - q^j - jpq^{j-1})^{m-\ell}.$$

Avec des considérations similaires à celles utilisées pour les non-redondants, il est possible de majorer la probabilité précédente par

$$\mathbb{P}[X \text{ est une traverse minimale}] \leq \frac{m!}{(m-j)!}(pq^{j-1})^j(1-q^j-jpq^{j-1})^{m-j}. \tag{5.3}$$

La proposition suivante s'appuie sur l'inégalité 5.3 et montre que le nombre moyen de traverses minimales est quasi-polynomial en le nombre de sommets et d'hyperarêtes.

Théorème 12 *Considérons le modèle $HG(n,m,p)$ avec $m=\beta n^\alpha$, $\beta>0$ et $\alpha>0$. Il existe une constante positive $c:=c(\alpha,\beta,p)$ telle que le nombre moyen de traverses minimales est*

$$\mathcal{O}\left(n^{d(\alpha)\log_{\frac{1}{q}} m + c\ln\ln m}\right),$$

avec $d(\alpha)=1$ si $\alpha\leq 1$ et $d(\alpha)=\frac{(\alpha+1)^2}{4\alpha}$ sinon.

Le théorème 12 donne une borne supérieure sur la taille moyenne de la sortie du problème THG de génération des traverses minimales. En combinant ce résultat avec l'inégalité de Markov, nous obtenons une borne supérieure presque sûre du nombre de traverses minimales dans un hypergraphe aléatoire. Afin d'étudier la complexité générique du problème THG, nous devons aussi trouver une borne inférieure presque sûre. Ce type de résultats s'obtient en étudiant les moments d'ordres supérieurs comme la variance mais nous ne sommes pas parvenus à mener à bien les calculs. Cependant, en reliant les traverses minimales aux traverses et en utilisant la propriété de concentration de la proposition 9-(v), nous obtenons la proposition suivante.

Proposition 11 *Soit $\epsilon\in]0,1[$. Dans le modèle aléatoire $HG(n,m,p)$, le nombre M de traverses minimales vérifie*

$$\mathbb{P}(M<\epsilon\mathbb{E}[T_\ell])=o(1)$$

avec T_ℓ l'ensemble des traverses de taille $\ell=\log_{\frac{1}{q}}\frac{mp}{\ln n}+1$. Rappelons que $\mathbb{E}[T_\ell]$ est donnée par la formule 5.1 et vérifie $\mathbb{E}[T_\ell]=\binom{n}{\ell}(1-q^\ell)^m$.

Encore une fois, la borne obtenue est quasi-polynomiale en m et n puisque nous avons

$$\mathbb{E}[T_\ell]=\exp\left(\frac{\ln(n)\cdot\ln(m)}{|\ln q|}(1+o(1))\right).$$

Nous disposons maintenant de tous les résultats pour montrer que l'algorithme MTMiner est presque sûrement output-polynomial dans le modèle $HG(m,n,p)$. Mais avant d'introduire l'algorithme $MTMiner$, nous présentons les résultats obtenus avec le modèle multivarié.

5.5 Résultats obtenus dans le modèle multivarié

Le modèle aléatoire $HG(n,m,g)$ donne un rôle différent à chaque sommet. Précisément, un sommet j appartient à une hyperarête avec une probabilité $p_j=g(j)$ qui lui est propre. Nous posons $q_j=1-p_j$. Dans ce modèle, nous avons distingué trois familles de sommets qui influencent différemment les asymptotiques.

- L'ensemble U des sommets ubiquitaires est formé des sommets $u\in V$ tels que $q_u<\frac{x}{m}$ pour une constante x fixée.
- L'ensemble R des sommets rares est tel que pour tout $r\in R$, $p_r<1-e^{-\frac{1}{\ln n}}$.
- Finalement, l'ensemble des autres sommets $S=V\setminus\{U\cup R\}$.

5.5.1 Borne inférieure sur le nombre de traverses

Nous posons μ_n la valeur moyenne des probabilités $q_1, q_2, \ldots, q_n$ soit

$$\mu_n = \frac{1}{n} \sum_{i=1}^{n} q_i.$$

Si $T(\mathcal{H})$ désigne le nombre de traverses dans l'hypergraphe $\mathcal{H}$, le nombre moyen de traverses dans le modèle $HG(m, n, g)$ est donné par

$$\mathbb{E}[T] = \sum_{\substack{X \subset V \\ |X| \geq 1}} (1 - \prod_{i \in X} q_i)^m.$$

En utilisant l'inégalité de Bernoulli $(1-t)^m \geq 1 - mt$ et l'inégalité géométrique, nous obtenons la proposition suivante.

Proposition 12 *Dans le modèle $HG(n, m, g)$, le nombre moyen de traverses est borné inférieurement par $2^n - m(1 + \mu_n)^n$.*

5.5.2 Borne supérieure sur le nombre de non-redondants

La proposition 12 donne indirectement une borne sur le nombre de non-redondants. En effet, il y a au plus $m(1 + \mu_n)^n$ ensembles de sommets qui ne sont pas des traverses et dans le pire des cas, ils sont tous non-redondants. Pour compter le nombre de non-redondants, il faut encore ajouter les traverses minimales. Mais chaque traverse minimale a un sous-ensemble strict qui est non-redondant et chaque non-redondant donne naissance a au plus n non-redondants. Nous obtenons ainsi le point 1 de la proposition suivante.

Proposition 13 *Dans le modèle $HG(n, m, g)$, le nombre de non-redondants vérifie les bornes suivantes.*

1. *Le nombre moyen de non-redondants (et donc de traverses minimales) est borné par $\mathcal{O}(mn(1 + \mu_n)^n)$.*
2. *Le nombre moyen de non-redondants contenant uniquement des sommets de $O \cup U$ est quasi-polynomial d'ordre $\mathcal{O}((nm \ln \sqrt{nm})^{\frac{1}{4} \ln n(\ln nm - 2 \ln \ln n - \ln \ln \sqrt{mn})})$.*
3. *Si $|R| = \mathcal{O}((\ln n)^c)$ avec c une constante, le nombre moyen de non-redondants est quasi-polynomial.*
4. *La probabilité d'avoir un nombre polynomial de non-redondants contenant au moins un sommet ubiquitaire tend vers 1.*

Le point 1 de la proposition 13 n'est pas très précis mais il peut facilement être interprété. Si μ_n tend vers 0, alors beaucoup de sommets sont présents dans presque toutes les hyperarêtes et il faut peu de sommets pour faire une traverse minimale. Si μ_n tend vers 1, alors les sommets ont tendance à devenir ubiquitaires et il faut combiner beaucoup plus de sommets pour obtenir des traverses minimales. Ceci explique le comportement exponentiel.

Les résultats 2 et 3 rendent plus précis le rôle asymptotique des évènements rares. Le nombre de non-redondants est exponentiel en le nombre de sommets rares et quasi-polynomial en les autres sommets. Si le nombre de sommets rares est d'ordre logarithmique, alors l'influence des sommets rares reste quasi-polynomial asymptotiquement. D'un autre côté, le résultat 4 montre que les sommets ubiquitaires ont un rôle polynomial dans l'asymptotique du nombre de non-redondants.

5.5.3 Nombre moyen de traverses minimales

Les traverses minimales étant des non-redondants, le théorème suivant est une conséquence directe de la proposition 13.

Théorème 13 *Dans le modèle $HG(n, m, g)$, le nombre M de traverses minimales vérifie les propriétés suivantes.*
- *Si $|S \cup R| = \mathcal{O}(\ln n)$, alors $\mathbb{E}[M]$ est au plus polynomial.*
- *Si $|R| = \mathcal{O}((\ln n)^c)$ avec c une constante, alors $\mathbb{E}[M]$ est au plus quasi-polynomial.*
- *Si $|R| = \Theta(n)$, alors $\mathbb{E}[M]$ est au plus exponentiel en $|R|$.*

5.6 Analyse de l'algorithme MTMiner

5.6.1 Présentation de l'algorithme et complexité moyenne

L'algorithme MTMiner a été créé par Hébert, Bretto et Crémilleux [75] en s'inspirant d'algorithmes de la fouille de données comme Apriori [21]. L'algorithme effectue un parcours en largeur du treillis des ensembles de sommets en appliquent des règles d'élagage à chaque niveau. Au niveau j, l'algorithme calcule les non-redondants formés de j sommets. Parmi les non-redondants, certains sont des traverses minimales et ils sont stockés dans une structure de données. Il est alors inutile de tester tous les sur-ensembles des traverses minimales trouvées car ils ne seront plus non-redondants. A l'inverse si un ensemble n'est pas redondant, il ne pourra pas être complété en une traverse minimale et il est aussi inutile de tester ses sur-ensembles. Les ensembles non-redondants restants de cardinal j sont utilisés pour construire les non-redondants de taille $j + 1$. MTMiner est décrit par l'algorithme 5.

Algorithme 5 : Algorithme MTMiner.

Entrée: Un hypergraphe $\mathcal{H} = \{V, \mathcal{E}\}$ avec n sommets et m hyperarêtes
Sortie: Les traverses minimales de $\mathcal{H}$

```
MT := {{v}| v ∈ V, {v} est une traverse}
N_1 := {{v}| v ∈ V \ MT, v appartient à au moins une hyperarête}
j = 1
tant que N_j ≠ ∅ faire
  pour tout préfixe P avec P ∪ {v_1}, P ∪ {v_2} ∈ N_j faire
    W = P ∪ {v_1} ∪ {v_2}
    si W est non-redondant alors
      si W est une traverse alors
        ajouter W à MT
      sinon
        ajouter W à N_{j+1}
      fin si
    fin si
  fin pour
  j = j + 1
fin tant que
retourner MT
```

Chaque non-redondant de taille j peut être étendu en au plus $n - j$ non-redondants de taille $j + 1$. Le test de chaque candidat peut se faire en temps polynomial et la complexité de MTMiner est alors d'ordre $O(\text{Poly}(m, n)N)$ avec N le nombre de non-redondants. En appliquant les propositions 10 et 13-3, nous obtenons les bornes de complexité moyenne suivantes.

Proposition 14 (*i*) *Soit j_0 donné par la proposition 10. Dans le modèle $HG(n,m,p)$, il existe une constante positive c' telle que la complexité moyenne de l'algorithme* MTMiner *est*

$$\mathcal{O}\left((m+n)^{c'}\binom{n}{j_0}\frac{m!}{(m-j_0)!}(pq^{j_0-1})^{j_0}\ln n\right).$$

(*ii*) *Dans le modèle $HG(n,m,g)$, si $|R| = \mathcal{O}((\ln n)^c)$ avec c une constante, alors la complexité moyenne de* MTMiner *est au plus quasi-polynomiale.*

Le théorème 12 montre que le nombre moyen de traverses minimales est quasi-polynomial en le nombre de sommets. La proposition 14 montre que la complexité moyenne de MTMiner est aussi quasi-polynomiale en le nombre de sommets avec un ordre de grandeur relié polynomialement au précédent. Ces résultats suggèrent que MTMiner est en moyenne output-polynomial mais ils ne suffisent pas à le démontrer. Il faut aussi s'assurer que la taille de la sortie n'est pas trop souvent trop petite par rapport à la complexité totale de l'algorithme. Pour cela, nous disposons de la borne inférieure presque-sûre sur le nombre de traverses minimales donnée par la proposition 11. En combinant ces résultats, nous obtenons le théorème suivant.

Théorème 14 *Considérons le modèle aléatoire $HG(n,m,p)$ avec $m = \beta n^\alpha$, $\beta > 0$ et $\alpha > 0$. Sous ce modèle, la complexité générique du problème* THG *de génération des traverses minimales est output-polynomiale. Précisément pour tout $\epsilon > 0$, l'algorithme* MTMiner *génère les traverses minimales d'un hypergraphe aléatoire en temps $\mathcal{O}\left((M+n)^{\epsilon+\frac{(\alpha+1)^2}{4\alpha}}\right)$ avec une probabilité asymptotiquement* 1 *où M est le nombre de traverses minimales.*

Autrement dit, le problème THG est presque sûrement output–polynomial. Ce résultat donne une réponse positive mais probabiliste au problème de savoir si THG est dans le pire des cas output-polynomial.

5.7 Perspectives

Dans ce chapitre nous avons adopté un point de vue probabiliste sur le problème de génération des traverses minimales. Nous avons analysé en moyenne les traverses, les non-redondants et les traverses minimales dans les deux modèles $HG(m,n,p)$ et $HG(m,n,g)$. Nous avons parfois obtenu des résultats précis mais aussi assez souvent des bornes supérieures qui sont toutes quasi-polynomiales. Nous avons aussi montré qu'en moyenne et à condition de ne pas avoir trop de sommets rares, générer les traverses minimales peut se faire presque sûrement en temps output-polynomial.

Les outils mathématiques utilisés pour prouver ces résultats relèvent surtout du dénombrement, de la méthode des moments et de calculs asymptotiques de sommes. Ces outils conduisent à des calculs qui ne contribuent pas à la bonne compréhension structurelle du problème. La combinatoire analytique s'appuie au contraire sur la structure du problème mais nous ne sommes pas parvenus à l'appliquer avec les traverses minimales. Pourtant plusieurs travaux mélant hypergraphes et combinatoire analytique existent. Par exemple, Ravelomanana et Laza Rijamamy [109] ont travaillé avec un modèle d'hypergraphes aléatoires dont la taille augmente avec le temps. Ce genre de modèle correspond à la démarche de certains algorithmes qui travaillent de manière itérative sur des hypergraphes de plus en plus grands. L'étude de la complexité moyenne du calcul des traverses minimales serait clairement intéressante dans ce contexte, surtout si la combinatoire analytique s'applique. De Panafieu [44] a aussi introduit un modèle riche

d'hypergraphes aléatoires et a appliqué la combinatoire analytique pour étudier l'évolution de la structure des hypergraphes. Là encore, analyser la complexité de génération des traverses minimales pourra peut-être se faire avec de la combinatoire analytique.

Ce chapitre s'inscrit plus largement dans le cadre de l'extraction de motifs dans les bases de données. Les traverses minimales correspondent aux motifs de la bordure négative. Pendant ma thèse, j'ai analysé le nombre moyen de motifs fréquents et fermés mais il en existe beaucoup d'autres. L'analyse de ces motifs constitue un projet de recherche à long terme mais palpitant de par les applications qui en découlent.

Conclusion

Dans ce mémoire, j'ai présenté plusieurs exemples d'analyses d'algorithmes dans trois domaines de l'informatique : les algorithmes du PGCD ou de réduction de réseaux euclidiens, la théorie de l'information et la fouille de données. Dans tous ces travaux, l'aléa est présent mais prend différentes formes.

Les travaux autour des algorithmes du PGCD et leurs généralisations (chapitres 2 et 3) sont dans la continuité de mes travaux de thèses. Des trois domaines cités plus haut, c'est très probablement celui que je maîtrise le mieux et il correspond un peu à *ma zone de confort*. Compte tenu de la diversité des algorithmes existants et non encore analysés, les projets de recherche ne manquent pas. Il me semble [6] que l'analyse en moyenne de certains algorithmes du PGCD en dimensions supérieures sont accessibles (Jacobi-Perron et Selmer par exemple). Un véritable défi sera de généraliser les techniques d'analyses en distribution développées par Baladi et Vallée pour deux entrées (avec des bornes à la Dolgopyat sur les opérateurs de transfert) à la dimension supérieure. Un autre axe de recherche serait l'étude des algorithmes du PGCD agissant sur d'autres ensembles que les polynômes et les entiers (entiers de Gauss par exemple). D'un point de vue plus méthodologique, nous avons mis en évidence des lois limites bêta. Ces lois sont rares en combinatoire analytique et il serait intéressant de caractériser des structures combinatoires qui conduisent à des lois bêta. Tous ces travaux contribueraient à étendre un peu plus le cadre applicatif de la combinatoire analytique ou de l'analyse dynamique.

Le chapitre 4 présente mes travaux en théorie de l'information. Avec Manuel Lladser, nous avons mis en évidence une loi limite gaussienne pour un motif caché très simple, composé de deux lettres, et donc sans autocorrélation. L'intérêt de ce travail réside surtout dans la méthode dont je pense qu'elle peut être étendue, dans un premier temps, à un motif à deux mots avec des corrélations simples. Mon apport le plus important en théorie de l'information réside plutôt dans le calcul explicite des taux d'entropie (relative) généralisés et renormalisés. Ces travaux sont le fruit d'une collaboration de près de 10 ans avec Valérie Girardin. Ce sont les connaissances acquises en analyse fonctionnelle et autour de la manipulation (formelle ou analytique) des séries génératrices qui ont contribué à l'aboutissement de ces travaux. Dans un travail en cours, nous projetons avec Valérie de donner une interprétation *plus* probabiliste de nos résultats. A plus long terme, nous prévoyons d'étendre nos travaux à des sources dont le taux d'entropie de Shannon est nul.

Pour terminer, mon travail de recherche en lien avec la fouille de données est présenté au chapitre 5. Ce travail sur les traverses minimales s'inscrit dans une démarche personnelle, commencée pendant ma thèse, qui vise à rapprocher la fouille de données et les analyses probabilistes d'algorithmes. Les deux domaines utilisent souvent les mêmes objets (graphes, hypergraphes, treillis, motifs, séquences, algorithmes) mais collaborent très peu entre eux. Je suis convaincu que les deux domaines ont un intérêt mutuel à collaborer ensemble. Les analyses probabilistes

6. Il faut être prudent car les analyses peuvent réserver des surprises désagréables...

peuvent aider à quantifier l'information dans les bases de données en comptant le nombre moyen de motifs par exemple. Elles peuvent aussi expliquer le comportement de certains algorithmes, voire optimiser le choix des algorithmes selon le contexte. D'un autre côté, la fouille de données impose des contraintes issues des applications qui nécessitent de concevoir des modèles aléatoires adaptés. Faire évoluer les méthodes de la combinatoire analytique pour intégrer ces modèles aléatoires est d'une part un challenge mathématique, et d'autre part viendra enrichir l'analyse d'algorithmes de nouvelles approches qui pourront probablement être réutilisées dans d'autres applications. Dans l'avenir, j'aimerais vraiment m'inscrire davantage dans cette démarche.

Bibliographie

[19] S. Abbes. The Information Rate of Asynchronous Sources. Dans *2nd International Conference on Information Communication Technologies*, volume 2, pages 3463–3467, 2006.

[20] D. Achlioptas et C. Moore. On the 2-Colorability of Random Hypergraphs. Dans *Proceedings of 6th RANDOM conference*, pages 78–90. Springer-Verlag, 2002.

[21] R. Agrawal et R. Srikant. Fast algorithms for mining association rules in large databases. Dans *Proceedings of the 20th International Conference on Very Large Data Bases*, VLDB '94, pages 487–499, San Francisco, CA, USA, 1994. Morgan Kaufmann Publishers Inc.

[22] M. Ajtai. The Shortest Vector Problem in L_2 is NP-hard for Randomized Reductions (Extended Abstract). Dans *STOC*, pages 10–19, 1998.

[23] M. Ajtai. Optimal lower bounds for the Korkine-Zolotareff parameters of a lattice and for Schnorr's algorithm for the shortest vector problem. *Theory of Computing*, 4(1) :21–51, 2008.

[24] A. Akhavi. Random lattices, threshold phenomena and efficient reduction algorithms . *Theoretical Computer Science*, 287(2) :359 – 385, 2002.

[25] A. Akhavi. The optimal LLL algorithm is still polynomial in fixed dimension. *Theoretical Computer Science*, 297(1-3) :3–23, 2003.

[26] A. Akhavi, J.-F. Marckert, et A. Rouault. On the Reduction of a Random Basis. Dans *Proceedings of the Meeting on Analytic Algorithmics and Combinatorics*, ANALCO '07, pages 265–270, Philadelphia, PA, USA, 2007. Society for Industrial and Applied Mathematics.

[27] A. Akhavi et B. Vallée. Average Bit-Complexity of Euclidean Algorithms. Dans *Proceedings of the 27th International Colloquium on Automata, Languages and Programming*, ICALP '00, pages 373–387, London, UK, UK, 2000. Springer-Verlag.

[28] D. Aldous. Probability Distributions on Cladograms. Dans *In Random Discrete Structures*, pages 1–18. Springer, 1996.

[29] V. Baladi et B. Vallée. Euclidean algorithms are Gaussian. *Journal of Number Theory*, 110 :331–386, 2006.

[30] M. Basseville. Divergence measures for statistical data processing—an annotated bibliography. *Signal Processing*, 93(4) :621 – 633, 2013.

[31] V. Berthé, J. Bourdon, T. Jolivet, et A. Siegel. Generating discrete planes with substitutions. Dans *Combinatorics on words*, volume 8079 of *LNCS*, pages 58–70. Springer, 2013.

[32] V. Berthé et H. Nakada. On Continued Fraction Expansions in Positive Characteristic : Equivalence Relations and some metric properties. *Expositiones Mathematicae*, 18 :257–284, 2000.

[33] J. Bourdon et B. Vallée. *Pattern Matching Statistics on Correlated Sources*, pages 224–237. Ed. Correa J. R. and Hevia A. and Kiwi M., Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[34] M. Bousquet-Mélou et K. Weller. Asymptotic Properties of Some Minor-Closed Classes of Graphs. *Combinatorics, Probability and Computing*, 23 :749–795, 9 2014.

[35] A. Broise-Alamichel et Y. Guivarc'h. Exposants caractéristiques de l'algorithme de Jacobi-Perron et de la transformation associée. *Annales de l'Institut Fourier*, 51(3) :565–686, 2001.

[36] V. Brun. Algorithmes euclidiens pour trois et quatre nombres. Dans *13e congrès des mathématiciens scandinaves, Helsinki 1957*, pages 45–64. 1958.

[37] F. Chazal et V. Maume-Deschamps. Statistical properties of general Markov dynamical sources : applications to information theory. *Discrete Mathematics & Theoretical Computer Science*, 6(2) :283–314, 2004.

[38] H. Chern et H. Hwang. Phase changes in random m-ary search trees and generalized quicksort. *Random Structure Algorithms*, 19(3-4) :316–358, 2001.

[39] H.-H. Chern, H.-K. Hwang, et T.-H. Tsai. An asymptotic theory for Cauchy–Euler differential equations with applications to the analysis of algorithms. *Journal of Algorithms*, 44(1) :177 – 225, 2002.

[40] T. M. Cover et J. A. Thomas. *Elements of Information Theory*. Wiley-Interscience, New York, NY, USA, 1991.

[41] P. Damaschke. Parameterized enumeration, transversals, and imperfect phylogeny reconstruction. *Theoretical Computer Science*, 351(3) :337–350, 2006.

[42] H. Daudé, P. Flajolet, et B. Vallée. An Average-Case Analysis of the Gaussian Algorithm for Lattice Reduction. *Combinatorics, Probability & Computing*, 6(4) :397–433, 1997.

[43] H. Daudé et B. Vallée. An Upper Bound on the Average Number of Iterations of the LLL Algorithm. *Theoretical Computer Science*, 123 :95–115, 1994.

[44] E. de Panafieu. Phase transition of random non-uniform hypergraphs. *Journal of Discrete Algorithms*, 31(0) :26 – 39, 2015. 24th International Workshop on Combinatorial Algorithms (IWOCA 2013).

[45] H. Delange. Généralisation du Théorème d'Ikehara. *Annales Scientifiques de l'ENS*, 71 :213–422, 1954.

[46] L. Devroye. Universal Limit Laws for Depths in Random Trees. *SIAM Journal on Computing*, 28(2) :409–432, 1998.

[47] J. D. Dixon. The number of steps in the euclidean algorithm. *Journal of Number Theory*, 2(4) :414 – 422, 1970.

[48] D. Dolgopyat. On decay of correlations in Anosov flows. *Annals of Mathematics*, 147(2) :357–390, 1998.

[49] D. Dolgopyat. Prevalence of rapid mixing in hyperbolic flows. *Ergodic Theory & Dynamical Systems*, 18 :1097–1114, 1998.

[50] A. Dudek et A. Frieze. Loose Hamilton Cycles in Random k-Uniform Hypergraphs, 2010.

[51] M. E. Dumitrescu. Some informational properties of Markov pure-jump processes. *Časopis pro pěstování matematiky*, 113(4) :429–434, 1988.

[52] T. Eiter et G. Gottlob. Identifying the Minimal Transversals of a Hypergraph and Related Problems. *SIAM Journal on Computing*, 24(6) :1278–1304, 1995.

[53] T. Eiter et G. Gottlob. Hypergraph Transversal Computation and Related Problems in Logic and AI. Dans *Proceedings 8th European Conference on Logics in Artificial Intelligence (JELIA 2002), LNCS/LNAI 2424* , pages 549–564, 2003.

[54] T. Eiter, G. Gottlob, et K. Makino. New Results on Monotone Dualization and Generating Hypergraph Transversals. *SIAM Journal on Computing*, 32(2) :514–537, 2003.

[55] T. Eiter, K. Makino, et G. Gottlob. Computational aspects of monotone dualization : A brief survey. *Discrete Applied Mathematics*, 156(11) :2035–2049, 2008.

[56] P. Erdős et A. Rényi. On random graphs. I. *Publicationes Mathematicae Debrecen*, 6 :290–297, 1959.

[57] P. Erdős et A. Rényi. On the Evolution of Random Graphs. Dans *Publication of the Mathematical Institute of the Hungarian Academy of Sciences*, pages 17–61, 1960.

[58] M. D. Esteban et D. Morales. A summary on entropy statistics. *Kybernetika*, 31(4) :337–346, 1995.

[59] V. Féray. Weighted dependency graphs. *ArXiv e-prints*, May 2016.

[60] P. Flajolet et R. Sedgewick. *Analytic Combinatorics*. Cambridge University Press, 2009.

[61] P. Flajolet, W. Szpankowski, et B. Vallée. Hidden Word Statistics. *Journal of the ACM*, 53(1) :147–183, 2006.

[62] M. L. Fredman et L. Khachiyan. On the Complexity of Dualization of Monotone Disjunctive Normal Forms. *Journal of Algorithms*, 21(3) :618–628, 1996.

[63] C. Friesen et D. Hensley. The statistics of continued fractions for polynomials over a finite field. *Proceedings of the American Mathematical Society*, 124 :2661–2673, 1996.

[64] M. Georgieva. *Analyse probabiliste de la réduction des réseaux euclidiens cryptographiques*. Thèse, Université de Caen, Dec. 2013.

[65] V. Girardin et N. Limnios. On the Entropy for Semi-Markov Processes. *Journal of Applied Probability*, 40(4) :1060–1068, 2003.

[66] V. Girardin et N. Limnios. Entropy Rate and Maximum Entropy Methods for Countable Semi-Markov Chains. *Communications in Statistics - Theory and Methods*, 33(3) :609–622, 2004.

[67] L. Golshani, E. Pasha, et G. Yari. Some properties of Rényi entropy and Rényi entropy rate. *Information Sciences*, 179(14) :2426 – 2433, 2009.

[68] L. Guibas et A. Odlyzko. String overlaps, pattern matching, and nontransitive games. *Journal of Combinatorial Theory, Series A*, 30(2) :183 – 208, 1981.

[69] L. J. Guibas et A. M. Odlyzko. Maximal Prefix-Synchronized Codes. *SIAM Journal on Applied Mathematics*, 35(2) :401–418, 1978.

[70] L. J. Guibas et A. M. Odlyzko. Periods in strings. *Journal of Combinatorial Theory, Series A*, 30(1) :19 – 42, 1981.

[71] D. Gunopulos, R. Khardon, H. Mannila, et H. Toivonen. Data mining, Hypergraph Transversals, and Machine Learning. Dans *Proceedings of the sixteenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems (PODS'97)*, pages 209–216, 1997.

[72] M. Hagen. *Algorithmic and Computational Complexity Issues of MONET*. Dissertation, Institut für Informatik, Friedrich-Schiller-Universität Jena, Dec. 2008.

[73] G. Hanrot, X. Pujol, et D. Stehlé. *Analyzing Blockwise Lattice Algorithms Using Dynamical Systems*, pages 447–464. Ed. Rogaway, Phillip, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[74] D. M. Hardcastle et K. Khanin. On almost everywhere strong convergence of multi-dimensional continued fraction algorithms. *Ergodic Theory Dynamical Systems*, 20(6) :1711–1733, 2000.

[75] C. Hébert, A. Bretto, et B. Crémilleux. A Data Mining Formalization to Improve Hypergraph Minimal Transversal Computation. *Fundamenta Informaticae*, 80 :415–433, December 2007.

[76] H. Heilbronn. On the average length of a class of finite continued fractions. Dans P. Turán, editeur, *Number Theory and Analysis*, pages 87–96. Springer US, 1969.

[77] P. Hennequin. *Analyse en moyenne d'algorithmes, tri rapide et arbres de recherche*. Thèse, École Polytechnique, 1991. Thèse de doctorat dirigée par Steyaert, Jean-Marc Sciences appliquées EP 1991.

[78] D. Hensley. The Number of Steps in the Euclidean Algorithm. *Journal of Number Theory*, 49(2) :142 – 182, 1994.

[79] H.-K. Hwang. On Convergence Rates in the Central Limit Theorems for Combinatorial Structures. *European Journal of Combinatorics*, 19(3) :329 – 343, 1998.

[80] H.-K. Hwang et V. Zacharovas. Limit laws of the coefficients of polynomials with only unit roots. *ArXiv e-prints*, Jan. 2013.

[81] I. Kapovich, A. Myasnikov, P. Schupp, et V. Shpilrain. Generic-case complexity, decision problems in group theory and random walks. *Journal of Algebra*, 264 :665–694, 2003.

[82] G. Kesidis et J. Walrand. Relative entropy between Markov transition rate matrices. *IEEE Transactions on Information Theory*, 39(3) :1056–1057, May 1993.

[83] A. Knopfmacher et J. Knopfmacher. The exact length of the Euclidean algorithm in $F_q[X]$. *Mathematika*, 35 :297–304, 1988.

[84] D. Knuth. The analysis of algorithms. *In Gauthier-Villars, Actes du congrès des mathématiciens*, (3) :269–274, 1971.

[85] D. Knuth. *The art or computer programming*, volume I–IV. Upper Saddle River, NJ : Addison-Wesley, 2006.

[86] A. Korkine et G. Zolotarev. Sur les formes quandratiques. *Mathematishe Annalen*, 6 :336–389, 1873.

[87] M. Kuba et A. Panholzer. Descendants in increasing trees. *The Electronic Journal of Combinatorics [electronic only]*, 13(1) :Research paper R8, 14 p.–Research paper R8, 14 p., 2006.

[88] S. Kullback et R. A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1) :79–86, March 1951.

[89] C. J. Lagarias. Geometric Models for Quasicrystals I. Delone Sets of Finite Type. *Discrete & Computational Geometry*, 21(2) :161–191, 1999.

[90] J. Lagarias, H. Lenstra, et C. Schnorr. Korkin-Zolotarev bases and successive minima of a lattice and its reciprocal lattice. *Combinatorica*, 10(4) :333–348, 1990.

[91] J. C. Lagarias. Worst-Case Complexity Bounds for Algorithms in the Theory of Integral Quadratic Forms. *Journal of Algorithms*, 1(2) :142–186, 1980.

[92] J. C. Lagarias. The quality of the diophantine approximations found by the jacobi-perron algorithm and related algorithms. *Monatshefte für Mathematik*, 115(4) :299–328, 1993.

[93] D. H. Lehmer. Euclid's Algorithm for Large Numbers. *The American Mathematical Monthly*, 45(4) :227–233, 1938.

[94] M. Lelarge. A new approach to the orientation of random hypergraphs. Dans *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '12, pages 251–264. SIAM, 2012.

[95] A. Lenstra, H. Lenstra, et L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4) :515–534, 1982.

[96] M.-F. M. Dimension and entropy of regular curves. Dans *Fractals, non-integral dimensions and applications*, pages 222–230. G. Cherbit ed., John Wiley and Sons, 1990.

[97] K. Ma et J. von zur Gathen. Analysis of Euclidean algorithms for polynomials over finite fields. *Journal of Symbolic Computation*, 9(4) :429 – 455, 1990.

[98] M. Madritsch et B. Vallée. Modelling the LLL algorithm by sandpiles. Dans *Proceedings of the 9th Latin American conference on Theoretical Informatics*, LATIN'10, pages 267–281, Berlin, Heidelberg, 2010. Springer-Verlag.

[99] H. Mahmoud. *Polya Urn Models*. Chapman & Hall/CRC, 1 edition, 2008.

[100] H. Mahmoud et R. T. Smythe. On the Distribution of Leaves in Rooted Subtrees of Recursive Trees. *The Annals of Applied Probability*, 1(3) :406–418, 1991.

[101] C. Martínez et S. Roura. Optimal Sampling Strategies in Quicksort and Quickselect. *SIAM J. Comput.*, 31(3) :683–705, Mar. 2002.

[102] M. L. Menéndez, D. Morales, L. Pardo, et M. Salicrú. (h, ϕ)-entropy differential metric. *Applications of Mathematics*, 42(2) :81–98, 1997.

[103] R. Neininger. On a multivariate contraction method for random recursive structures with applications to Quicksort. *Random Structure Algorithms*, 19(3-4) :498–524, 2001.

[104] P. Q. Nguyen et B. Vallée. *The LLL Algorithm : Survey and Applications*. Springer Publishing Company, Incorporated, 1st edition, 2009.

[105] P. Nicodème, B. Salvy, et P. Flajolet. Motif statistics. *Theoretical Computer Science*, 287(2) :593 – 617, 2002.

[106] Z. Rached, F. Alajaji, et L. Campbell. Rényi's Entropy Rate For Discrete Markov Sources. Dans *Proceedings of the CISS'99*, pages 17–19, Baltimore, MD, 1999.

[107] Z. Rached, F. Alajaji, et L. L. Campbell. Rényi's divergence and entropy rates for finite alphabet Markov sources. *IEEE Transactions on Information Theory*, 47(4) :1553–1561, May 2001.

[108] Z. Rached, F. Alajaji, et L. L. Campbell. The Kullback-Leibler divergence rate between Markov sources. *IEEE Transactions on Information Theory*, 50(5) :917–921, May 2004.

[109] V. Ravelomanana et A. Rijamamy. Creation and Growth of Components in a Random Hypergraph Process. Dans D. Chen et D. Lee, editeurs, *Computing and Combinatorics*, volume 4112 of *Lecture Notes in Computer Science*, pages 350–359. Springer Berlin Heidelberg, 2006.

[110] M. Régnier et W. Szpankowski. On the Approximate Pattern Occurrences in a Text. Dans *Proceedings of the Compression and Complexity of Sequences 1997*, SEQUENCES '97, pages 253–264, Washington, DC, USA, 1997. IEEE Computer Society.

[111] M. Régnier et W. Szpankowski. On Pattern Frequency Occurrences in a Markovian Sequence. *Algorithmica*, 22(4) :631–649, 1998.

[112] G. J. Rieger. Über die mittlere Schrittanzahl bei Divisionalgorithmen. *Mathematische Nachrichten*, pages 157–180, 1978.

[113] D. Ruelle. *Dynamical Zeta Functions for Piecewise Monotone Maps of the Interval*. CRM monograph series. American Mathematical Society, 2006.

[114] A. Rényi. On measures of entropy and information. Dans *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1 : Contributions to the Theory of Statistics*, pages 547–561, Berkeley, California, 1961. University of California Press.

[115] M. Salicru, M. Menendez, D. Morales, et L. Pardo. Asymptotic distribution of (h, φ)-entropies. *Communications in Statistics-Theory and Methods*, 22(7) :2015–2031, 1993.

[116] A. P. Sant'anna et I. J. Taneja. Trigonometric entropies, Jensen difference divergence measures, and error bounds. *Information Sciences*, 35(2) :145 – 156, 1985.

[117] S. Saswati et S. Kumar N. Hypergraph models for cellular mobile communication systems. *IEEE Transactions on Vehicular Technology*, 47(2) :460–471, 1998.

[118] C. P. Schnorr. A hierarchy of polynomial time lattice basis reduction algorithms. *Theoretical Computer Science*, 53(2-3) :201–224, Aug. 1987.

[119] A. Schönhage. Schnelle berechnung von kettenbruchentwicklungen. *Acta Informatica*, 1(2) :139–144, 1971.

[120] F. Schweiger. *Multidimensional continued fractions*. Oxford University Press, 2000.

[121] R. Sedgewick. The analysis of quicksort programs. *Acta Inf.*, 7 :327–355, 1977.

[122] I. Semaev. *A 3-Dimensional Lattice Reduction Algorithm*, pages 181–193. Ed. Silverman, Joseph H., Springer Berlin Heidelberg, Berlin, Heidelberg, 2001.

[123] C. Shannon. A Mathematical Theory of Communication. *Bell System Technical Journal*, 27 :379–423, 623–656, 1948.

[124] B. Sharma et P. Mittal. New non-additive measures of relative information. *Journal of Combinatorics, Information & System Sciences*, 2 :122–133, 1975.

[125] I. Shmulevich, A. D. Korshunov, et J. Astola. Almost all monotone Boolean functions are polynomially learnable using membership queries. *Information Processing Letters*, 79(5) :211–213, Sept. 2001.

[126] W. Stummer et I. Vajda. On Bregman Distances and Divergences of Probability Measures. *IEEE Transactions on Information Theory*, 58(3) :1277–1288, March 2012.

[127] I. J. Taneja, L. Pardo, D. Morales, et M. L. Menéndez. On generalized information and divergence measures and their applications : a brief review. *Qüestiió*, 13(1,2,3) :47–73, 1989.

[128] C. Tsallis. Possible generalization of Boltzmann-Gibbs statistics. *Journal of Statistical Physics*, 52(1) :479–487, 1988.

[129] A. Ullah. *Entropy Divergence and Distance Measures with Econometric Applications*. Working paper in economics. Department of Economics, University of California, Riverside, 1993.

[130] B. Vallée. *An affine point of view on minima finding in integer lattices of lower dimensions*, pages 376–378. Ed. Davenport, James H., Springer Berlin Heidelberg, 1989.

[131] B. Vallée. Gauss' Algorithm Revisited. *Journal of Algorithms*, 12(4) :556–572, Dec. 1991.

[132] B. Vallée. Dynamical Sources in Information Theory : Fundamental Intervals and Word Prefixes. *Algorithmica*, 29(1/2) :262–306, 2001.

[133] B. Vallée. Euclidean Dynamics. *Discrete and Continuous Dynamical Systems*, 1(15) :281–352, 2006.

[134] B. Vallée et A. Vera. Lattice reduction in two dimensions : analyses under realistic probabilistic models. Dans P. Jacquet, editeur, *2007 Conference on Analysis of Algorithms, AofA 07*, volume AH of *DMTCS Proceedings*, pages 197–234, Juan les Pins, France, 2007. Discrete Mathematics and Theoretical Computer Science.

[135] B. Vallée. Opérateurs de Ruelle-Mayer généralisés et analyse en moyenne des algorithmes d'Euclide et de Gauss. *Acta Arithmetica*, 81(2) :101–144, 1997.

[136] B. Vallée. Dynamical analysis of a class of Euclidean algorithms. *Theoretical Computer Science*, 297(1) :447 – 486, 2003.

[137] A. Vera. *Analyses de l'algorithme de Gauss. Applications à l'analyse de l'algorithme LLL.* Thèse, University of Caen, 2009.

[138] S. Wild, M. E. Nebel, et H. Mahmoud. Analysis of Quickselect under Yaroslavskiy's Dual-Pivoting Algorithm. *CoRR*, abs/1306.3819, 2013.

[139] S. Wild, M. E. Nebel, et R. Neininger. Average Case and Distributional Analysis of Java 7's Dual Pivot Quicksort. *CoRR*, abs/1304.0988, 2013.

Résumé

Introduit par Knuth dans les années 60, l'analyse d'algorithmes est un domaine de l'Informatique Mathématique qui vise à comprendre le comportement des algorithmes lorsque la taille des entrées augmente. Le plus souvent, les analyses concernent la complexité en temps ou en mémoire mais d'autres paramètres peuvent aussi être étudiés. Les analyses dans le pire et le meilleur des cas se focalisent sur les comportements extrémaux des paramètres qui correspondent généralement à des entrées pathologiques rares dans les applications. Les analyses d'algorithmes tiennent compte de toutes les entrées et attribuent un poids à chacune d'entre elles. Dans l'idéal, une entrée plus fréquente dans les applications a un poids plus important. Ainsi, les analyses décrivent plus fidèlement le comportement *typique* des algorithmes.

Depuis les années 60, de nombreuses méthodes ont été proposées. Nous en utilisons deux : la Combinatoire Analytique développée en France depuis les années 80 autour de Philippe Flajolet, et l'Analyse Dynamique, développée autour de Brigitte Vallée depuis le milieu des années 90. Ces méthodes reposent sur des manipulations à la fois formelles et analytiques de séries génératrices. L'Analyse Dynamique s'appuie en plus sur des opérateurs fonctionnels issus de l'étude des systèmes dynamiques.

Les algorithmes étudiés dans ce mémoire sont en lien avec trois domaines de l'informatique : les algorithmes du PGCD, la Théorie de l'Information et la Fouille de Données. Après s'être intéressés aux algorithmes calculant le PGCD de deux entiers ou polynômes, nous nous orientons maintenant vers des généralisations à la dimension supérieure. Ce mémoire présente les résultats obtenus pour deux types de généralisations possibles : les algorithmes qui calculent le PGCD de plusieurs entiers ou polynômes et les algorithmes de réduction de réseaux euclidiens. Dans le cadre de la Théorie de l'Information, nous étudions une grandeur caractéristique des sources : les taux d'entropie. Nous nous intéressons aussi à la fréquence d'une famille de motifs dans un texte produit par une source. Finalement, nous étudions la complexité d'extraire les motifs de la bordure négative en Fouille de données, ou de manière équivalente, à la complexité de générer l'hypergraphe transversal d'un hypergraphe aléatoire.

Bien que les domaines et les objets abordés soient très différents, l'aléa et l'analyse d'algorithmes structurent tous les travaux présents dans ce mémoire.

Mots-clés: Analyses d'algorithmes, combinatoire analytique, analyse dynamique, algorithmes du PGCD, réduction de réseaux euclidiens, Théorie de l'Information, taux d'entropies, extraction de motifs dans les bases de données, hypergraphes.