Journée annuelle
2011, p. 49 – 70

LE PSEUDO-ALÉA : OBJETS ET GÉNÉRATION

par

David Xiao

***Résumé.* —** L'aléa est un outil indispensable dans l'informatique théorique. Il trouve aussi de plus en plus d'applications dans les preuves mathématiques. Malgré son utilité, il est souvent souhaitable de réduire ou même d'éliminer l'utilisation de l'aléa. Nous verrons qu'il est souvent possible de remplacer l'aléa par des objets pseudo-aléatoires, et nous examinerons plusieurs exemples : les graphes expandeurs, les extracteurs d'aléa, les générateurs pseudo-aléatoires, et les codes correcteurs d'erreurs. Nous exposerons aussi le principe de transfert, qui permet de transférer des propriétés d'un ensemble pseudo-aléatoire à ses sous-ensembles.

***Abstract.* —** Randomness is indispensable in theoretical computer science, and has found progressively more applications in proofs in mathematics. Despite its usefulness, it is often desirable to reduce or eliminate randomness when possible. We will see that it is often possible to replace randomness with pseudorandom objects. We will look at several examples: expander graphs, extractors, pseudorandom generators, and error-correcting codes. We will also discuss the transference principle, which allows us to transfer properties from pseudo-random sets to their subsets.

1. Introduction

1.1. Le pseudo-aléa et l'informatique. — L'aléa est une ressource de calcul très importante. Il permet d'accélérer le calcul de certains algorithmes, comme les tests de primalité de Miller-Rabin [**Mil75, Rab80**] et de Solovay-Strassen [**SS77**], qui ont été pendant plus que vingt ans les seuls tests de primalité connus. Plus récemment, Agrawal *et al.* [**AKS02**] ont montré qu'il est possible de tester la primalité en temps polynomial déterministe, mais les tests probabilistes de Miller-Rabin et Solovay-Strassen restent plus efficaces en pratique. L'aléa permet aussi de résoudre certains problèmes pour lesquels aucun algorithme efficace et déterministe n'est connu. Par exemple, le seul algorithme efficace connu pour le problème suivant est probabiliste.

© Journée annuelle, SMF 2011

1.1.1. Exemple : test d'identité des polynômes.— On se donne un polynôme $p \in \mathbb{F}[x_1, \ldots, x_n]$ de degré d à coefficients dans un corps fini $\mathbb{F}$ de taille $|\mathbb{F}| \geq 2d$. On doit accepter le polynôme si $p \equiv 0$, et sinon le rejeter. Le polynôme peut prendre une forme complexe, par exemple $(x_1x_2 - 2x_3)^7(x_4x_5 - x_2^2) + (x_3 - x_5)^9$, et il peut alors contenir un nombre de coefficients qui est exponentiel en le degré d. Alors, l'idée triviale, qui consiste à développer tous les coefficients, n'est pas satisfaisante, car elle ne donne pas lieu à un algorithme efficace, *i.e.* un algorithme qui se termine en temps polynomial. On décrit dans l'Algorithme 1.1 un algorithme probabiliste qui résout ce problème en temps polynomial.

***Théorème 1.1* ([Sch80, Zip79]).** — *L'Algorithme 1.1 accepte tout polynôme $p \equiv 0$ avec probabilité 1. Il rejette tout polynôme $p \not\equiv 0$ avec probabilité $\geq 1/2$. L'algorithme se termine en temps polynomial.*

Nous prouverons ce théorème dans la Section 2.2. On peut constater qu'il est facile de réduire l'erreur dans le cas où $p \not\equiv 0$. Il suffit simplement de répéter l'algorithme probabiliste, en utilisant de l'aléa indépendant à chaque répétition. Si on répète k fois, l'erreur est réduite à 2^{-k}, alors que la quantité d'aléa nécessaire croît linéairement avec le nombre de répétitions.

En général, on considère que l'aléa est une ressource coûteuse. Les philosophes émettent depuis toujours des doutes sur l'existence de l'aléa ; les informaticiens supposent que l'aléa existe, mais cherchent à l'économiser autant que possible. Ils cherchent aussi à se munir d'outils pour utiliser de l'aléa défectueux (c'est-à-dire non-uniforme), car l'aléa existant peut être non-uniforme.

Nous verrons plusieurs méthodes pour économiser l'aléa. Nous verrons que, grâce aux *graphes expandeurs*, on peut réduire l'erreur dans les algorithmes probabilistes sans aucun aléa supplémentaire. Nous verrons enfin que les *extracteurs d'aléa* permettent de transformer une source d'aléa avec une distribution quelconque, pourvu qu'elle ait suffisamment d'entropie, en une source d'aléa uniforme.

La possibilité d'économiser l'aléa nous mène à nous demander si l'aléa est nécessaire dans ces algorithmes. Autrement dit, est-il possible de montrer que tout problème résoluble avec un algorithme probabiliste et efficace peut être résolu par un algorithme déterministe et efficace ? Aujourd'hui, on sait que, sous certaines conjectures, la réponse est affirmative : sous ces conjectures, on peut construire des *générateurs pseudo-aléatoires* qui permettent de rendre déterministe (*i.e.* dérandomiser) tout algorithme probabiliste et efficace. L'idée est de rendre la dépendance vis-à-vis de l'aléa si faible que l' énumération de tous les choix possible d'aléa prenne un temps polynomial.

Entrée : Un polynôme $p \in \mathbb{F}[x_1, \ldots, x_n]$ de degré d. Soit $S \subseteq \mathbb{F}$ un sous-ensemble de taille $2d$ quelconque.

1. Tirer au hasard $z_1, \ldots, z_n \in S$.
2. Accepter p si $p(z_1, \ldots, z_n) = 0$, sinon rejeter p.

Algorithm 1. 1. — *Algorithme probabiliste et efficace pour tester les polynômes.*

1.2. Le pseudo-aléa et les mathématiques. — L'étude de l'aléa et, plus généralement des probabilités en mathématiques, remonte au XVII$^{\text{e}}$ siècle, mais l'idée du pseudo-aléa est assez récente. Le pseudo-aléa s'applique le plus souvent dans le cas où l'aléa est utilisé pour montrer l'existence de certains objets. Cette méthode, souvent appelée la *méthode probabiliste*, a été mise en valeur surtout par Erdős. De manière informelle, la méthode probabiliste est la suivante : pour montrer qu'un objet possédant une certaine propriété existe, on définit un ensemble d'objets et on montre qu'un objet choisi au hasard dans cet ensemble vérifie la propriété avec une probabilité non-nulle. Nous en donnons un exemple simple :

1.2.1. Exemple : les graphes de Ramsey.— La théorie de Ramsey étudie à partir de quelle taille un objet peut jouir de certaines propriétés. C'est le théorème de Ramsey [**Ram30**] qui est à l'origine de cette théorie. Ce théorème considère les cliques dans les graphes. Pour un graphe $G = (V, E)$, on désigne par $n = |V|$ le nombre de ses sommets, et on définit le complément de G comme $\overline{G} = (V, \overline{E})$ où $(i, j) \in \overline{E}$ ssi $(i, j) \notin E$. Le théorème de Ramsey affirme que, pour tout graphe G, avec un nombre n de sommets suffisamment grand, un des deux graphes G ou $\overline{G}$ contient une clique de taille $\Omega(\log n)$.

Pour tout t donné, le nombre de Ramsey $R(t)$ est le plus petit entier n tel que, pour tout graphe G à n sommets, soit G soit $\overline{G}$ contienne une clique de taille t. Le théorème de Ramsey montre la borne supérieure $R(t) = 2^{O(t)}$. Erdős a donné une borne inférieure en utilisant la méthode probabiliste [**Erd47**]. Ici nous donnons une version faible de la borne de Erdős.

***Théorème 1.2* ([Erd47]). —** *Pour t suffisamment grand, il existe un graphe G à $2^{t/2}$ sommets pour lequel ni G ni $\overline{G}$ ne contiennent aucune clique à t sommets.*

Nous prouverons ce théorème dans la Section 2.2. On peut en déduire que $R(t) > 2^{t/2}$. Depuis la preuve de Erdős, qui est non constructive, les mathématiciens recherchent une preuve constructive de ce théorème : ils cherchent à construire explicitement un graphe G tel que ni G ni $\overline{G}$ ne contienne aucune clique à t sommets. Dans l'esprit algorithmique, on dit qu'une preuve est *constructive* ou *explicite* si elle donne lieu à un algorithme déterministe et efficace qui construit l'objet cherché.

Malheureusement, les meilleurs graphes explicites connus sont loin d'atteindre le rapport exponentiel $n = 2^{\Omega(t)}$ donné par le Théorème 1.2 ; pendant longtemps, la meilleure borne constructive était due à Frankl-Wilson [**FW81**]. Plus récemment, Barak *et al.* ont amélioré cette borne en utilisant des idées issues de l'informatique [**BRSW06**].

Malgré notre incapacité à donner une construction explicite du Théorème 1.2, il existe d'autres problèmes pour lesquels nous savons construire des objets pseudo-aléatoires qui se comportent presque comme des objets aléatoires. Nous verrons l'exemple des codes correcteurs d'erreurs.

Finalement, nous examinerons une autre application de l'idée du pseudo-aléa : le principe de transfert. Le principe de transfert affirme (en gros) que, si un ensemble est pseudo-aléatoire, alors ses sous-ensembles sont aussi pseudo-aléatoires. Il s'avère que ce principe s'applique à l'ensemble des nombres premiers, et c'est l'un des ingrédients

principaux de la preuve du théorème de Green-Tao [**GT08**], qui affirme que l'ensemble des nombres premiers contient des suites arithmétiques arbitrairement longues.

1.3. L'objet de cet exposé. — Dans cet exposé, nous présenterons les idées de base de l'étude du pseudo-aléa, à travers des exemples concrets. Il existe un schéma général formel qui englobe la plupart des objets pseudo-aléatoires décrits dans la suite de l'exposé. Pourtant, ce schéma est assez abstrait, alors que les objets eux-mêmes sont plutôt concrets et simples à décrire. De l'avis de l'auteur, ce qui rapproche ces objets de manière plus évidente mais aussi édifiante, c'est le fait que les techniques utilisées pour les construire et les comprendre sont très souvent les mêmes : les groupes et les corps finis, l'algèbre linéaire, l'entropie, la borne de Chernoff, etc.

Il existe d'ailleurs plusieurs textes qui exposent l'histoire et les motivations de l'étude du pseudo-aléa. Nous les citons à la fin de l'exposé dans les conseils de lecture, et nous conseillons vivement au lecteur de les consulter. Afin que cet exposé soit complémentaire de ces textes cités, nous privilégions ici des exemples, plutôt que l'histoire et les motivations du domaine.

Pour ces raisons, nous choisissons dans cet exposé de traiter des objets pseudo-aléatoires individuels, de manière détaillée, avec parfois des démonstrations complètes. Nous n'insisterons pas trop sur les liens entre ces différents objets sauf quand le lien est évident. Dans la conclusion, nous décrirons un point de vue global sur le pseudo-aléa. Nous espérons que, après avoir vu quelques exemples de ces techniques, le lecteur sera bien préparé à aller plus loin, et à découvrir et comprendre d'autres exemples du pseudo-aléa.

Il existe des notions liées au pseudo-aléa que nous ne traiterons pas ici, faute de place : la dérandomisation par les espérances conditionnelles, les fonctions de hachage universelles, l'indépendance par k-uplets, etc. À la fin de l'exposé, nous donnerons des conseils de lecture qui traitent ces sujets ainsi que d'autres.

Pour privilégier la simplicité et la clarté des explications, les énoncés de la plupart des théorèmes cités dans cet exposé ne sont pas optimaux d'un point de vue quantitatif.

Nous commençons par donner les définitions utiles ; mais, comme nous n'utiliserons pas de définitions très compliquées, nous invitons le lecteur qui le souhaite à passer directement aux sections suivantes et à retourner à la Section 2, seulement s'il en éprouve le besoin.

2. Préliminaires

2.1. Notation. — Nous supposons que le lecteur a des notions de base en probabilité. Nous travaillons uniquement avec des espaces de probabilité discrets, et, dans la plupart des cas, avec des espaces de probabilité finis. Nous rappelons la borne de Chernoff, qui va intervenir plusieurs fois dans cet exposé :

Théorème 2.1 **(Borne de Chernoff).** — *Si $X_1, \dots, X_k$ sont des distributions indépendantes avec support dans $[0,1]$, et si $\mu = \mathbb{E}[\frac{1}{k}\sum_{i=1}^{k} X_i]$, alors, pour tout $\varepsilon \in]0,1[$, on*

a l'inégalité suivante :

$$\Pr\left[\left|\frac{1}{k}\sum_{i=1}^{k} X_i - \mu\right| > \varepsilon\right] \leq 2e^{-\varepsilon^2 k/3}$$

Soit X une distribution. Nous écrivons $x \leftarrow_{\mathrm{R}} X$ pour désigner une variable aléatoire x tirée de la distribution X. Si S est un ensemble fini, S désigne aussi la distribution uniforme sur l'ensemble S. Pour un entier positif n, on utilise la notation $[n] = \{1, \ldots, n\}$, et $\{0,1\}^n$ désigne l'espace de chaînes de bits de longueur n ; la chaîne de bits de longueur n où chaque bit vaut 1 est désignée par 1^n. U_n désigne la distribution uniforme sur $\{0,1\}^n$. Si $z \in \{0,1\}^*$, $|z|$ désigne la longueur de z. Pour un ensemble S et un entier k qui vérifie $0 \leq k \leq |S|$, la notation $\binom{S}{k}$ désigne l'ensemble $\{T \mid T \subseteq \ S\text{et } |T| = k\}$.

Pour mesurer la distance entre deux distributions X et Y qui prennent leurs valeurs dans un même univers $\mathcal{U}$, nous utilisons *la distance de variation totale*

$$\Delta(X,Y) = \max_{T \subseteq \mathcal{U}} |\Pr[X \in T] - \Pr[Y \in T]|$$

2.1.1. Définitions algorithmiques. — Si A est un algorithme déterministe, on dit que A est efficace s'il se termine en temps polynomial, c'est-à-dire s'il existe une constante c, telle que, pour toute entrée $z \in \{0,1\}^n$, le calcul de $A(z)$ se termine après un nombre d'étapes au plus égal à n^c.

Un algorithme probabiliste est un algorithme déterministe qui prend comme entrée auxilliaire une chaîne de bits aléatoire ω. On dit qu'un algorithme probabiliste A est efficace s'il existe une constante c telle que, pour toute entrée $z \in \{0,1\}^n$ et tout choix d'aléa ω, le calcul $A(z;\omega)$ se termine en temps n^c. Remarquons que dans ce cas on peut supposer sans perte de généralité que $|\omega| \leq n^c$.

Pour un langage $L \subseteq \{0,1\}^*$, on écrit $L(z) = 1$ si $z \in L$ et $L(z) = 0$ autrement. On dit qu'un algorithme probabiliste A *décide* de l'appartenance à L avec erreur bornée par $\delta(n)$ si :

$$\forall n, \forall z \in \{0,1\}^n,\ \Pr_{\omega}[A(z;\omega) \neq L(z)] \leq \delta(n)$$

On peut facilement réduire l'erreur en répétant l'algorithme k fois avec des choix d'aléa ω indépendants, et en prenant la majorité des $A(z;\omega_1), \ldots, A(z;\omega_k)$. La borne de Chernoff dit que l'erreur diminue alors de δ à $2e^{-(\frac{1}{2}-\delta)^2 k/3}$.

2.2. Preuves d'échauffement. — Nous prouvons maintenant les théorèmes énoncés dans l'introduction. Ces preuves sont des exemples très simples du style de raisonnement typique dans ce domaine.

Preuve du Théorème 1.1. — Le fait que l'Algorithme 1.1 se termine en temps polynomial est évident. Le fait qu'il accepte tout polynôme $p \equiv 0$ est aussi évident. Il reste à montrer qu'il rejette tout polynôme $p \not\equiv 0$. Nous prouvons le lemme suivant, qui implique immédiatement le théorème.

***Lemme 2.2* ([Sch80, Zip79]).** — *Pour tout $p \in \mathbb{F}[x_1, \ldots, x_n]$ de degré d qui vérifie $p \not\equiv 0$, pour tout ensemble fini $S \subseteq \mathbb{F}$, on a l'inégalité :*

$$\Pr_{z_1,\ldots,z_n \leftarrow_R S}[p(z_1, \ldots, z_n) = 0] \leq \frac{d}{|S|}.$$

Le lemme est vérifié pour $n = 1$ parce que le nombre de racines d'un polynôme non-nul à une indéterminée et de degré d est inférieur ou égal à d. Supposons maintenant que le lemme est vérifié pour $n-1$ indéterminées et prouvons le lemme pour n indéterminées.

Le polynôme p peut s'écrire sous la forme

$$p(x_1, \ldots, x_n) = \sum_{j=1}^{d} x_1^j q_j(x_2, \ldots, x_{n-1})$$

où les q_j sont des polynômes à $n-1$ indéterminées, dont le degré est inférieur ou égal à $d-j$. Si $p \not\equiv 0$, il existe k maximal tel que $q_k \not\equiv 0$.

Regardons maintenant le choix de $z_2, \ldots, z_n$. L'hypothèse de récurrence montre l'inégalité

$$\Pr_{z_2,\ldots,z_n \leftarrow_R S}[q_k(z_2, \ldots, z_n) = 0] \leq \frac{d-k}{|S|}. \tag{2.1}$$

D'autre part, si $q_k(z_2, \ldots, z_n) \neq 0$, le polynôme $p(x_1, z_2, \ldots, z_n)$ est un polynôme non-nul à une indéterminée, de degré au plus k, et on peut écrire

$$\Pr_{z_1,\ldots,z_n \leftarrow_R S}[p(z_1, \ldots, z_n) = 0 \mid q_k(z_2, \ldots, z_n) \neq 0] \leq \frac{k}{|S|} \tag{2.2}$$

En combinant l'Inégalité 2.1 et l'Inégalité 2.2, nous obtenons

$$\begin{aligned}
\Pr_{z_1,\ldots,z_n \leftarrow_R S}[p(z_1, \ldots, z_n) = 0] &= \Pr[p(z_1, \ldots, z_n) = 0 \wedge q_k(z_2, \ldots, z_n) = 0] \\
&\quad + \Pr[p(z_1, \ldots, z_n) = 0 \wedge q_k(z_2, \ldots, z_n) \neq 0] \\
&\leq \Pr[q_k(z_2, \ldots, z_n) = 0] \\
&\quad + \Pr[p(z_1, \ldots, z_n) = 0 \mid q_k(z_2, \ldots, z_n) \neq 0] \\
&\leq \frac{d}{|S|}
\end{aligned}$$

■

Preuve du Théorème 1.2.. — Posons $n = 2^{t/2}$ et désignons par $\mathcal{G}_n = \{G = (V, E) \mid n = |V|\}$ l'ensemble de tous les graphes à n sommets. Pour montrer l'existence d'un graphe qui satisfait l'énoncé, on choisit au hasard un graphe G dans l'ensemble $\mathcal{G}_n$, et on montre que la probabilité que G ou $\overline{G}$ contienne une clique de taille t est faible.

Remarquons que, pour tout $u, v \in V$ distincts, l'arête (u, v) apparaît dans un graphe choisi de $\mathcal{G}_n$ avec probabilité $1/2$. Alors, pour tout $S \in \binom{[n]}{t}$, la probabilité que S soit une clique dans G ou dans $\overline{G}$ est $2^{-\binom{t}{2}+1}$. Désignons par $\mathsf{Clique}(S, G)$

l'événement « S est une clique dans G ou dans $\overline{G}$ ». La borne de l'union permet d'écrire :

$$\Pr_{G \leftarrow_{\mathrm{R}} \mathcal{G}_n}\left[\exists S \in \binom{[n]}{t}, \mathsf{Clique}(S, G)\right] \leq \binom{n}{t} 2^{-\binom{t}{2}+1}$$

Pour $n = 2^{t/2}$, cette dernière probabilité tend vers 0 pour t suffisamment grand. On en déduit l'existence d'un graphe qui satisfait l'énoncé. ∎

3. Les graphes expandeurs

Les graphes expandeurs sont des graphes où chaque sommet est « très bien connecté » à tous les autres sommets. Il y a plusieurs critères importants dans la définition de la notion de « très bonne connexité ». Par exemple, on veut qu'une marche aléatoire sur le graphe converge très vite vers la distribution uniforme sur les sommets. On veut aussi qu'il n'existe pas de coupes très éparses, parce que de telles coupes correspondraient à des « embouteillages » dans le graphe. On peut aussi exiger que le voisinage de chaque sous-ensemble de sommets soit beaucoup plus grand que le sous-ensemble lui-même.

Il est trivial de bien connecter tous les sommets en prenant un graphe complet, mais pour les graphes expandeurs on exige que le degré de chaque sommet soit borné par un paramètre D, ce qui veut dire que le nombre d'arêtes est borné par $\frac{nD}{2}$ au lieu de $\binom{n}{2}$. Dans le cas le plus intéressant, D est constant.

3.1. Définition et propriétés. — Il y a plusieurs façons de formaliser la notion d'un graphe « très bien connecté », et chaque définition formalise un des critères intuitifs évoqués ci-dessus. Toutes les définitions sont à peu près équivalentes, et nous allons travailler uniquement avec l'une d'entre elles. L'exposé de Hoory *et al.* [**HLW06**] est une excellente référence, qui traite tous les aspects des graphes expandeurs, et nous invitons le lecteur à le consulter pour l'énoncé d'autres définitions d'expansion, les démonstrations d'équivalence entre ces définitions, et une exposition de beaucoup d'autres résultats sur les graphes expandeurs.

La définition avec laquelle on travaille ici est une définition spectrale. Dans cette partie de l'exposé, nous considérons uniquement les graphes non-orientés et connexes. Nous identifions d'une manière arbitraire les sommets d'un graphe à n sommets avec les entiers $[n] = \{1, \ldots, n\}$. Nous nous intéressons aux graphes réguliers, où chaque sommet a le même nombre de voisins.

Pour un graphe $G = (V, E)$ qui est D-régulier, nous définissons la matrice d'adjacence normalisée $M = [m_{ij}]_{i,j \in [n]}$ telle que $m_{ij} = \frac{1}{D}$ si $(i, j) \in E$ et $m_{ij} = 0$ autrement. Comme le graphe est non-orienté, la matrice M est symétrique, positive, et doublement stochastique. Le théorème spectral prouve alors que M possède n valeurs propres réelles qu'on ordonne dans l'ordre décroissant : $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$. Le théorème spectral affirme aussi que les vecteurs propres correspondants $v_1, \ldots, v_n$ forment une base orthonormale de $\mathbb{R}^n$. Pour les graphes connexes, on peut d'ailleurs montrer que $\lambda_i \in [-1, 1]$ pour tout $1 \leq i \leq n$, que la plus grande valeur propre vérifie

$\lambda_1 = 1$, et qu'après normalisation, le vecteur propre v_1 est le vecteur uniforme où chaque composante vaut $1/\sqrt{n}$.

L'écart entre les valeurs propres contrôle l'expansion d'un graphe. En particulier, si $\max\{|\lambda_2|, |\lambda_n|\}$ est strictement plus petit que $\lambda_1 = 1$, le graphe est un graphe expandeur. Nous définissons aussi les familles de graphes expandeurs.

***Définition 3.1* (Graphes expandeurs).** — Posons $\lambda(G) = \max\{|\lambda_2|, |\lambda_n|\}$.

1. Un graphe G à n sommets est un (n, D, α)-*graphe expandeur* s'il est D-régulier et vérifie $\lambda(G) \leq \alpha$.
2. Une famille de graphes $\mathcal{G}$ est une (D, α)-*famille de graphes expandeurs* si chaque $G \in \mathcal{G}$ à n sommets est un (n, D, α)-graphe expandeur.

On s'intéresse surtout aux (D, α)-familles de graphes expandeurs pour lesquelles les paramètres D, α sont constants. Nous regardons maintenant quelques propriétés de tels graphes vis-à-vis de critères liés au pseudo-aléa.

3.1.1. La convergence des marches aléatoires. — La proposition suivante dit que les graphes de la Définition 3.1 satisfont un des critères intuitifs de « très bonne connexité » : une marche aléatoire sur un tel graphe converge vite vers la distribution uniforme.

Proposition 3.2. — *Soit G un (n, D, α) graphe expandeur. La distribution du k-ième sommet visité par une marche aléatoire sur G au départ d'un sommet quelconque est $\alpha^k\sqrt{n}/2$-proche de la distribution uniforme vis-à-vis de la distance de variation totale.*

Nous esquissons la démonstration. On peut identifier toute distribution X sur V à un vecteur $x \in \mathbb{R}^n$, où la composante x_i est définie par $x_i = \Pr[X = i]$. De plus, tout vecteur $x \in \mathbb{R}^n$ peut se décomposer dans la base orthonormale des vecteurs propres $v_1, \ldots, v_n$ de la matrice d'adjacence M de G.

Soit e_i le vecteur dont la i-ième composante vaut 1 et les autres composantes sont nulles. Ce vecteur e_i représente la distribution constante sur le sommet de départ i, et e_i se décompose sur la base (v_j) en $\sum_{j=1}^n a_j v_j$. On vérifie que $a_1 = 1$. Si la marche commence à i, la distribution qui résulte après un pas aléatoire dans le graphe G est donnée par le vecteur Me_i, et

$$Me_i = M\sum_{j=1}^n a_j v_j = v_1 + \sum_{j=2}^n a_j \lambda_j v_j.$$

La composante selon v_1 reste inchangée, mais les composantes selon les autres vecteurs $v_2, \ldots, v_n$ sont multiplées par λ_j qui vérifie $|\lambda_j| \leq \alpha < 1$. Si on applique maintenant M^k, les composantes selon les vecteurs $v_2, \ldots, v_n$ sont multipliées par λ_j^k avec $|\lambda_j|^k \leq \alpha^k$ et deviennent insignifiantes, et alors $M^k e_i$ converge vers le vecteur v_1. Le premier vecteur propre v_1 correspond à la distribution uniforme, et on en déduit que la distribution $M^k e_i$ converge vers la distribution uniforme.

Cette proposition montre que pour tout $\varepsilon > 0$, toute marche marche aléatoire au départ d'un sommet quelconque, arrive, après $O(\log(n/\varepsilon)/\log(1/\alpha))$ pas, à une distribution qui est ε-proche de la distribution uniforme. Remarquons que ceci est

optimal à une constante près pour les graphes de degré constant, car le diamètre de tels graphes est $\Omega(\log n)$.

3.1.2. La concentration des marches aléatoires. — On peut montrer que les marches aléatoires sur un graphe expandeur non seulement convergent vers la distribution uniforme, mais que l'ensemble des sommets visités pendant la marche constitue lui aussi un ensemble presque aléatoire. C'est-à-dire, pour tout sous-ensemble de sommets $S \subseteq V$, la fraction des sommets visités qui sont dans S est proche de $|S|/|V|$.

Théorème 3.3. — *Soit G un (n, D, α)-graphe expandeur. Soit $S \subseteq [n]$ un sous-ensemble des sommets dont la cardinalité est de la forme $|S| = \mu n$. Soit un entier $k > 0$, et soit une marche aléatoire W de longueur k sur G, qui part d'un sommet choisi au hasard. La suite $X_1, \ldots, X_k$ est définie par les égalités suivantes : $X_i = 1$ si le i-ième sommet visité par W est dans S, et $X_i = 0$ autrement. Alors, pour tout $\varepsilon \in]0, 1[$, on a :*

$$\Pr\left[\left|\frac{1}{k}\sum_{i=1}^{k} X_k - \mu\right| > \varepsilon\right] \leq 2e^{-\frac{\varepsilon(1-\alpha)k}{4}}$$

Ce théorème a d'abord été prouvé (à des constants près) par Gillman [**Gil93**]. La version citée ci-dessus est due à Healy [**Hea06**].

Ce théorème généralise la borne de Chernoff. La borne de Chernoff prouve qu' une suite de tirages indépendants permet de réduire exponentiellement l'erreur. Le Théorème 3.3 montre que la suite des tirages indépendants peut être remplacée par une marche aléatoire sur un graphe expandeur, ce qui est moins coûteux en terme d'aléa utilisés. (voir la Section 3.3.2).

3.1.3. Le lemme de mélange. — Dans un graphe aléatoire D-régulier, le nombre d'arêtes $E(S, T)$ entre deux sous-ensembles $S, T \subseteq V$ quelconques est défini par

$$E(S, T) = |\{(i, j) \in E \mid i \in S, j \in T\}|.$$

Le lemme suivant montre que, dans un graphe expandeur, le nombre d'arêtes $E(S, T)$ est assez proche de sa valeur moyenne, égale à $(1/n)D|S||T|$

***Lemme 3.4* ([AC88]).** — *Soit G un (n, D, α) graphe expandeur. Pour tout $S, T \subseteq V$*

$$\left|E(S, T) - \frac{D|S| \cdot |T|}{n}\right| \leq \alpha D\sqrt{|S| \cdot |T|}$$

La démonstration du lemme est fondée sur la décomposition des vecteurs dans la base des vecteurs propres de M, déjà évoquée dans la discussion de la Proposition 3.2.

3.2. Constructions. — La méthode probabiliste permet de montrer l'existence de (D, α)-familles de graphes expandeurs, correspondant à des paramètres D, α constants. Il faut donc d'abord définir une distribution adéquate sur les graphes D-réguliers (voir par exemple chapitre 7 de [**HLW06**]).

Pour utiliser les graphes expandeurs, il faut non seulement prouver l'existence de tels graphes, mais élaborer un algorithme qui les construit de manière explicite. On dit qu'un (n, D, α)-graphe expandeur G est *explicite* si, pour tout sommet $i \in G$ et

tout $k \in [D]$, on peut calculer le k-ième voisin de i (selon un ordre arbitraire sur les voisins de i) en temps polynomial en $\lceil \log n \rceil$. Remarquons que le temps est polynomial en $\lceil \log n \rceil$ parce seuls $\lceil \log n \rceil$ bits sont nécessaires pour décrire un sommet.

Aujourd'hui, on connaît plusieurs familles de graphes expandeurs explicites. Un des plus simples à décrire est due à Gabber et Galil :

Définition 3.5. — Soit $\mathbb{Z}/n\mathbb{Z}$ l'anneau des entiers modulo n. Le graphe de Gabber-Galil sur $2n$ sommets est défini ainsi : son ensemble de sommets est $V = (\mathbb{Z}/n\mathbb{Z})^2$ et chaque sommet $(x, y) \in V$ est relié aux huit sommets suivants :

$$\begin{array}{cccc} (x, y+2x) & (x, y+2x+1) & (x, y-2x) & (x, y-2x-1) \\ (x+2y, y) & (x+2y+1, y) & (x-2y, y) & (x-2y-1, y) \end{array}$$

***Théorème 3.6* ([GG81, JM87]).** — *Les graphes de Gabber-Galil forment une* $(8, \frac{5\sqrt{2}}{8})$*-famille de graphes expandeurs explicite.*

Nous invitons le lecteur à consulter [**GG81, JM87**] pour deux démonstrations différentes de ce théorème (voir aussi [**Xia03**] et chapitre 8 de [**HLW06**] pour des expositions de ces preuves).

On peut montrer que les paramètres (D, α) d'un (n, D, α) graphe expandeur doivent satisfaire l'inégalité $\alpha \geq \frac{2\sqrt{D-1}}{D}$ [**Nil91**]. Remarquons que les graphes de Gabber-Galil n'atteignent pas cette borne. Il existe cependant des familles de graphes expandeurs explicites qui atteignent cette borne, dont la construction est due à Lubotzky *et al.* [**LPS88**] et à Margulis [**Mar88**]. Ces graphes sont définis en utilisant la théorie des groupes et des représentations, et sont souvent appelés les graphes de Ramanujan.

Il existe aussi des familles de graphes expandeurs explicites qui ne sont pas liées à la théorie des groupes et qui sont définies de manière complètement combinatoire. La première telle construction est due à Reingold *et al.* [**RVW02**]. Les auteurs définissent et utilisent le *produit zig-zag*, un produit de graphes qui permet d'augmenter le nombre de sommets tout en préservant la propriété d'expansion.

3.3. Applications. — Nous montrons ici comment les graphes expandeurs peuvent aider à réduire la quantité d'aléa utilisée dans les algorithmes probabilistes. Supposons qu'un algorithme probabiliste et efficace A décide un langage L avec erreur bornée par $1/3$. On a vu dans la Section 2.1.1 qu'il est possible de réduire l'erreur jusqu'à $2^{-\Omega(k)}$ en prenant la majorité de k répétitions indépendantes.

Le désavantage de cette méthode réside dans la quantité d'aléa utilisée. Si, au départ, on utilisait m bits d'aléa pour une seule exécution de l'algorithme avec une erreur de $1/3$, alors, en prenant des répétitions indépendantes, on doit utiliser km bits d'aléa pour obtenir une erreur de $2^{-\Omega(k)}$. Puisqu'on considère que l'aléa « coûte cher », on voudrait, si possible, réduire l'erreur sans « dépenser » autant d'aléa.

3.3.1. Réduction déterministe d'erreur. — On montre qu'on peut réduire l'erreur sans dépenser *aucun* aléa supplémentaire :

***Théorème 3.7* ([CW89]).** — *Soit L un langage. S'il existe un algorithme probabiliste et efficace A qui décide de l'appartenance à L avec erreur $1/3$ en utilisant m bits d'aléa,*

alors, pour tout $c > 0$, il existe un algorithme probabiliste et efficace A' qui décide de l'appartenance à L avec erreur $1/n^c$ en utilisant m bits d'aléa.

Démonstration. — Soit $\mathcal{G}$ une (D, α)-famille de graphes expandeurs explicites, avec des paramètres D, α constants. Soit $k = O_{\alpha,c}(\log(n))$ de sorte que $12\alpha^{2k} \leq 1/n^c$. Remarquons que $D^k = n^{O_{\alpha,c}(1)}$ est polynomial en n.

Supposons, pour simplifier, qu'il existe un graphe $G \in \mathcal{G}$ de taille 2^m. Par définition, le graphe G^k a le même ensemble de sommets que G et (i, j) est une arête dans G^k si et seulement s'il existe un chemin de longueur k de i à j dans G (on permet des arêtes multiples dans G^k s'il existe plusieurs chemins entre i et j dans G). La matrice d'adjacence de G^k est M^k, et on peut alors vérifier que G^k est un $(2^m, D^k, \alpha^k)$-graphe expandeur explicite.

Identifions les sommets de G^k avec $\{0,1\}^m$. L'algorithme A' fonctionne comme suit sur l'entrée z :

1. Tirer $\omega_0 \leftarrow_{\mathrm{R}} \{0,1\}^m$. Soient $\omega_1, \ldots, \omega_{D^k}$ tous les voisins de ω_0 dans G^k.
2. Calculer $A(z;\omega_1), \ldots, A(z;\omega_{D^k})$.
3. Retourner la réponse majoritaire.

L'algorithme A' n'utilise que m bits d'aléa parce que le seul choix aléatoire est ω_0. Il est efficace parce que G^k est explicite, $D^k = n^{O(1)}$ et que chaque exécution de A est efficace.

Il reste à montrer que l'erreur est bornée par $1/n^c$. Fixons l'entrée z. Soit $B = \{\omega \mid A(z;\omega) \neq L(z)\}$, l'ensemble des aléas qui mène A à une mauvaise réponse sur l'entrée z. Puisque l'erreur de A est bornée par $1/3$, on en déduit que $|B| \leq 2^m/3$.

Soit $B' = \{\omega \mid A'(z;\omega) \neq L(z)\}$, l'ensemble des aléas qui mènent A' à une mauvaise réponse sur l'entrée z. Par définition de A', et pour chaque $\omega \in B'$, la majorité des voisins de ω est dans B. Cela implique que $E(B, B') \geq |B'|D^k/2$.

On applique alors le lemme de mélange, Lemme 3.4 :

$$\begin{aligned}
\left|\frac{E(B,B')}{D^k} - \frac{|B|\cdot|B'|}{2^m}\right| &\leq \alpha^k\sqrt{|B|\cdot|B'|} \\
\frac{D^k|B'|}{2D^k} - \frac{|B|\cdot|B'|}{2^m} &\leq \alpha^k\sqrt{|B|\cdot|B'|} \\
|B'| &\leq \frac{\alpha^{2k}|B|}{\left(\frac{1}{2} - \frac{|B|}{2^m}\right)^2} \\
|B'| &\leq 12\alpha^{2k}2^m
\end{aligned}$$

Grâce au choix de k, on en déduit que $|B'| \leq 2^m/n^c$, et alors la probabilité d'erreur est donc bornée par $1/n^c$. ∎

3.3.2. Réduction exponentielle d'erreur. — Il est aussi possible d'obtenir une réduction exponentielle de l'erreur en économisant de l'aléa avec des graphes expandeurs. Pour cela, on identifie les sommets d'un graphe expandeur avec $\{0,1\}^m$, l'espace des chaînes d'aléa utilisé par l'algorithme, comme dans la preuve du théorème Théorème 3.7. On effectue une marche aléatoire dans le graphe expandeur, et, pour chaque sommet visité, on applique l'algorithme de départ à la chaîne de bits qui correspond au

sommet, et on retourne la réponse majoritaire. L'ensemble B des sommets pour lesquels l'algorithme de départ donne la mauvaise réponse a une cardinalité qui vérifie $|B| \leq 2^m/3$. Si le graphe est un $(2^m, D, \alpha)$ graphe expandeur, le Théorème 3.3 montre que la probabilité que la majorité des sommets visités pendant une marche aléatoire donne la mauvaise réponse est bornée par $2e^{-((1-\alpha)k/144)}$. Cet algorithme dépense $r + k \log D$ bits d'aléa, beaucoup moins que les rk bits dépensés avec des répétitions indépendentes.

4. Les extracteurs d'aléa

On suppose que l'aléa existe dans la nature, par exemple dans les phénomènes quantiques, mais, en pratique, cet aléa comporte toujours des erreurs et du bruit. Il semble donc trop optimiste de croire que cet aléa naturel puisse être parfaitement uniforme, c'est-à-dire qu'on puisse y trouver des bits qui sont 0 ou 1 avec probabilité exactement 1/2. Pourtant, dans l'analyse des algorithmes probabilistes, on suppose toujours que l'aléa de l'algorithme est parfaitement uniforme. Si on veut implémenter ces algorithmes en pratique, on doit montrer comment substituer l'aléa défectueux disponible dans la réalité à l'aléa idéal dont on avait supposé l'existence dans les preuves.

En pratique, les programmateurs utilisent comme graine l'heure actuelle, et ils y appliquent une fonction comme le LFSR (*linear feedback shift register*) et ils supposent que le résultat est suffisament « aléatoire » pour pouvoir être utilisé dans un algorithme probabiliste. Pourtant, l'heure actuelle a très peu d'aléa, et le LFSR n'est pas pseudo-aléatoire.

Pour combler cet écart entre aléa défectueux et aléa uniforme, nous introduisons les *extracteurs d'aléa*.

4.1. Définition. — Avant de définir ce qu' est un extracteur d'aléa, on fait quelques remarques sur les conditions que doit satisfaire la source d'aléa défectueux de départ. En l'occurrence, toutes les sources défectueuses ne peuvent pas être transformées en source presque uniforme. Il est nécessaire par exemple que l'aléa de départ ait suffisamment d'entropie, et on travaille ici avec la définition d'entropie suivante :

Définition 4.1. — Soit X une distribution qui prend ses valeurs dans un univers $\mathcal{U}$. La *min-entropie* de X est $H_\infty(X) = \min_{u \in \mathcal{U}} |\log \Pr[X = u]|$.

Dans ce contexte, la min-entropie est mieux adaptée que l'entropie de Shannon, qui est l'entropie habituelle, définie comme $H(X) = \sum_{u \in \mathcal{U}} \Pr[X = u] |\log \Pr[X = u]|$. Pour comprendre pourquoi l'entropie de Shannon ne suffit pas, considérons la distribution X qui prend ses valeurs dans $\{0,1\}^n$ et qui vaut 1^n avec probabilité 0.99 et qui vaut une chaîne uniformément aléatoire avec probabilité 0.01. Dans ce cas, l'entropie de Shannon est grande, elle vérifie $H(X) = \Omega(n)$, mais il est clair que X ne peut pas être transformer en aléa uniforme, puisqu'elle est constante avec une probabilité presque égale à 1.

À partir d'une source X qui prend ses valeurs dans $\{0,1\}^n$ bits, et qui a suffisamment d'entropie $H_\infty(X) \geq k$, on peut espérer extraire k bits suffisamment uniformes.

Pourtant, si $k < n$, même si on ne veut extraire qu'un seul bit, et même si on a suffisamment d'entropie, il n'est pas toujours possible d'utiliser une seule transformation. C'est facile à vérifier : à partir d' une fonction $f : \{0,1\}^n \to \{0,1\}$ quelconque, on montre l'existence d'une source S avec min-entropie $n-1$, telle que $f(S)$ soit constante. Considérons les pre-images $f^{-1}(0), f^{-1}(1)$, supposons sans perte de généralité que $|f^{-1}(0)| \geq |f^{-1}(1)|$, et considérons une distribution X, uniforme $f^{-1}(0)$. Alors $H_\infty(X) \geq n-1$ mais $\Pr[f(X)=0]=1$.

On peut éviter ce problème en permettant l'accès de la transformation f à une *graine* aléatoire de longueur s :

Définition 4.2. — Une fonction $f : \{0,1\}^n \times \{0,1\}^s \to \{0,1\}^m$ est un (k,ε)-*extracteur d'aléa* si, pour toute source X vérifiant $H_\infty(X) \geq k$, on a

$$\Delta(f(X,U_s),\ U_m) \leq \varepsilon$$

La graine dans la définition est supposée parfaitement aléatoire. On peut se demander pourquoi on se permet une telle graine si le but à l'origine était de travailler avec de l'aléa défectueux. On verra qu'il est possible de construire des extracteurs d'aléa où la taille de la graine est courte, $s = O(\log n)$, et on peut ainsi éliminer l'aléa de la graine en essayant toutes les valeurs possibles. En général, on préfère que la graine de l'extracteur soit courte et que la sortie soit longue.

4.2. Construction. — On peut montrer l'existence des extracteurs d'aléa en choisissant une fonction $f : \{0,1\}^n \times \{0,1\}^s \to \{0,1\}^m$ au hasard :

Théorème 4.3. — *Pour tout n, ε et $k \leq n$, il existe des (k,ε)-extracteurs d'aléa avec une graine de longueur $s = \log((n-k)/\varepsilon^2) + O(1)$, et une sortie de longueur $m = k + d - 2\log(1/\varepsilon) - O(1)$.*

Nous laissons la preuve au lecteur. Les paramètres du Théorème 4.3 sont optimaux à des constantes additives près [**RTS00**].

Pour utiliser les extracteurs dans des algorithmes, il faut construire des extracteurs explicites, c'est-à-dire qui sont calculables au moyen d'algorithmes efficaces. Ceci est possible avec des paramètres optimaux à des constantes multiplicatives près.

Théorème 4.4 **([GUV09])**. — *Pour tout n, k, tout $\gamma > 0$ et tout $\varepsilon > \exp(-n/2^{O(\log^* n)})$, il existe un (k,ε) extracteur explicite avec graine de longueur $s = O(\log(n/\varepsilon))$ et sortie de longueur $m = (1-\gamma)k$.*

La construction de cet extracteur utilise deux ingrédients principaux : les codes correcteurs d'erreurs de Parvaresh-Vardy, et des graphes expandeurs.

Nous présentons une construction simple d'extracteur d'aléa qui est presque optimal quand l'entropie k est grande. Cet extracteur est basé sur les marches aléatoires sur les graphes expandeurs.

Théorème 4.5 **([Zuc07])**. — *Pour tout $\gamma, \varepsilon > 0$, il existe $\beta > 0$ et un $((1-\beta)n, \varepsilon)$ extracteur explicite $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ avec $m \geq (1-\gamma)n$ et $s = \log(\gamma n/3)$.*

Démonstration. — Considérons la famille des graphes de Gabber-Galil (voir Définition 3.5), qui constitue une $(8,\alpha)$-famille de graphes expandeurs avec $\alpha = \frac{5\sqrt{2}}{8}$ (Théorème 3.6), et travaillons avec le graphe G de Gabber-Galil sur 2^m sommets.

L'extracteur E prend son entrée $x \in \{0,1\}^n$ et l'identifie avec une marche aléatoire de longueur 2^s sur G : en effet, grâce au choix de m et s, on a $n = m + 3 \cdot 2^s$, et les m premiers bits de x définissent un sommet dans G tandis que les $3 \cdot 2^s$ bits suivants décrivent les choix de voisins dans une marche aléatoire sur G. Soit y la graine de l'extracteur. On peut interpréter y comme un entier dans $[2^s]$. La sortie de E est le y-ième sommet visité par la marche définie par x.

Cette fonction est efficacement calculable, puisque les graphes de Gabber-Galil sont explicites. Pour montrer que cela définit un (k,ε) extracteur pour $k = (1-\beta)n$, il faut prouver que, pour toute source X avec $H_\infty(X) \geq k$, l'inégalité $\Delta(E(X,U_s),U_m) \leq \varepsilon$ est vérifiée.

Dans la construction des extracteurs, l'astuce consiste à prouver l'énoncé pour les sources plates, c'est-à-dire les sources qui sont uniformes sur un sous-ensemble $X \subseteq \{0,1\}^n$ de taille $|X| \geq 2^k$. C'est suffisant car toute source avec k bits de min-entropie peut se décomposer en une combinaison convexe de sources plates, qui ont chacune k bits de min-entropie.

En utilisant cette astuce, il suffit de montrer que pour tout partie $X \subseteq \{0,1\}^n$ de taille $|X| \geq 2^k$, la distance vérifie $\Delta(E(X,U_s),U_m) \leq \varepsilon$. La définition de la distance de variation totale entraîne

$$\Delta(E(X,U_s),U_m) = \max_{S \subseteq \{0,1\}^m} |\Pr[E(X,U_s) \in S] - |S|2^{-m}| \tag{4.1}$$

$$= \max_{S \subseteq \{0,1\}^m} \left| \frac{1}{X} \sum_{x \in X} \Pr_{U_s}[E(x,U_s) \in S] - |S|2^{-m} \right|. \tag{4.2}$$

La quantité $\Pr[E(x,U_s) \in S]$ est simplement la fraction des pas de la marche définie par x qui sont dans S. On divise $\{0,1\}^n$ en deux ensembles : T est l'ensemble des x pour lesquels la fraction des sommets dans la marche définie par x qui sont dans S est dans l'intervalle $[\frac{|S|}{2^m} - \frac{\varepsilon}{2}, \frac{|S|}{2^m} + \frac{\varepsilon}{2}]$, et son complémentaire $\overline{T} = \{0,1\}^n \setminus T$.

Par définition, chaque sommet $x \in T$ ne contribue que $\frac{\varepsilon}{2}$ à la distance. Chaque sommet $x \notin T$ peut y contribuer pour au plus 1, et on peut ainsi écrire

$$\Delta(E(X,U_s),U_m) \leq \max_{S \subseteq \{0,1\}^m} \frac{|\overline{T}|}{|X|} + \frac{\varepsilon}{2} \tag{4.3}$$

On peut appliquer le Théorème 3.3 pour montrer que si x est choisi au hasard, la probabilité que $x \in \overline{T}$ est bornée par $2e^{-\varepsilon^2(1-\alpha)2^s/16}$. On en déduit que $|\overline{T}| \leq 2^n \cdot 2e^{-\varepsilon^2(1-\alpha)2^s/16}$.

On applique ce fait et le fait que $|X| \geq 2^k = 2^{(1-\beta)n}$ à l'Inégalité 4.3 :

$$\Delta(E(X,U_s),U_m) \leq 2^{\beta n} 2^{-\varepsilon^2(1-\alpha)2^s/16} + \frac{\varepsilon}{2} \tag{4.4}$$

Si on choisit $\beta < \varepsilon^2(1-\alpha)\gamma/48$, on peut calculer que la distance est bornée par ε pour n suffisament grand. ∎

4.3. Applications. — Les extracteurs d'aléa permettent d'utiliser une source d'aléas défectueux dans des algorithmes qui supposent un accès à une source d'aléa uniforme. L'idée est simple : à partir d'un algorithme probabiliste A, et d'un accès à une source X avec min-entropie suffisante, on tire un échantillon $x \leftarrow_{\mathrm{R}} X$, on prend un extracteur explicite E comme celui du Théorème 4.4, et on calcule $\omega_y = E(x, y)$ pour toutes les valeurs possibles de y. Comme il n'existe qu'un nombre polynomial de y, le calcul est efficace. On calcule A sur tous les ω_y, et on retourne la réponse majoritaire.

5. Les générateurs pseudo-aléatoires

Jusqu'ici, nous avons vu comment réduire la dépendance vis-à-vis de l'aléa uniforme dans les algorithmes probabilistes. Nous avons expliqué qu'il est possible de réduire l'erreur des algorithmes probabilistes sans dépenser plus d'aléa, et qu'il est possible de gérer l'aléa défectueux. On peut aller plus loin et poser la question : Est-il possible d'éliminer l'aléa complètement ? Autrement dit, est-il possible de rendre déterministe n'importe quel algorithme probabiliste ?

La démarche naturelle passe par la construction d' un *générateur pseudo-aléatoire*. Un générateur pseudo-aléatoire (souvent appelé PRG pour la dénomination anglaise « pseudo-random generator ») est une fonction qui étire une graine de bits aléatoires très courte en une longue chaîne de bits pseudo-aléatoires. Comme pour les extracteurs d'aléa, l'idée est de construire des PRG avec des graines si courtes qu' on pourra remplacer le tirage d' une graine aléatoire, par l' énumération exhaustive de toutes les graines possibles.

La définition du pseudo-aléatoire est fondée sur la similitude du comportement d'un algorithme probabiliste et efficace, quand il utilise un l'aléa choisi uniformément ou quand il utilise un aléa produit par le générateur :

Définition 5.1. — Un (s, m, ε)-*générateur pseudo-aléatoire* est une fonction $G : \{0,1\}^s \to \{0,1\}^m$ calculable en temps polynomial en m tel que pour tout algorithme efficace et probabiliste A qui utilise m bits d'aléa :

$$\forall z \in \{0,1\}^n,\ \left|\Pr[A(z; G(U_s)) = 1] - \Pr[A(z; U_m) = 1]\right| \leq \varepsilon$$

Remarquons que l'efficacité de G est définie par rapport à la longueur m de sa sortie et non pas par rapport à la longueur de son entrée. Cette définition d'efficacité est plus adaptée car nous allons souvent choisir $s = O(\log m)$.

5.1. La dérandomisation. — En utilisant la définition précédente du générateur pseudo-aléatoire, on peut facilement montrer que l'existence de certains PRG rend possible la dérandomisation de tout algorithme probabiliste efficace.

Proposition 5.2. — *S'il existe un* $(O(\log n), n^c, 1/10)$*-PRG, alors, pour tout langage* L *dont l'appartenance est décidable au moyen d'un algorithme probabiliste et efficace qui utilise* n^c *bits avec erreur* $1/3$*, on peut décider de l'appartenance de* L *au moyen d' un algorithme déterministe efficace.*

Démonstration. — La construction de l 'algorithme déterministe A' s'effectue simplement à partir de l'algorithme probabiliste A de départ. Pour une entrée z fixée, on calcule $A(z; G(x))$ pour toutes les valeurs possibles de $x \in \{0,1\}^{O(\log n)}$ et on retourne la réponse majoritaire. L'algorithme A' est efficace parce que A, G sont efficaces et qu'il existe $n^{O(1)}$ choix possibles de x. L'algorithme A' donne la bonne réponse : comme G est un PRG avec erreur $1/10$, alors la définition du PRG implique que A ne peut pas distinguer la sortie de G de la distribution uniforme, et ainsi la réponse majoritaire est correcte. ∎

Pour dérandomiser tout algorithme efficace, il suffit alors donc de disposer de PRG, mais malheureusement on ne connaît pas de construction explicite de ces objets. Pourtant, comme on va le montrer dans la suite, Nisan et Wigderson ont montré qu'il est possible de construire des PRG *sous la conjecture qu'il existe des fonctions difficiles à calculer*. Nous remarquons ici que la réciproque est également vraie : l'existence des PRG, même celle d'un PRG qui dérandomise seulement l'algorithme 1.1 qui résout le problème du test des polynômes, implique l'existence des fonctions difficiles à calculer [**KI03**].

5.1.1. Le PRG de Nisan-Wigderson. — Nisan et Wigderson [**NW94**] établissent un lien étroit entre la difficulté de calcul et les PRG. On donne la définition suivante : on dit qu' une fonction $f : \{0,1\}^n \to \{0,1\}$ est difficile à calculer (on suppose pour simplifier que f est équilibrée, $\Pr_{x \leftarrow_R \{0,1\}^n}[f(x) = 1] = 1/2$), si pour tout algorithme déterministe et efficace A,

$$\left| \Pr_{x \leftarrow_R \{0,1\}^n}[A(x) = f(x)] - \tfrac{1}{2} \right| \leq \varepsilon$$

On en déduit que, si f est difficile, il est aussi difficile de distinguer $(x, f(x))$ de (x, b) pour un bit b indépendant et uniforme. Donc, dans un sens à préciser, la fonction $x \mapsto (x, f(x))$ est un PRG. Nisan et Wigderson montrent alors que si on part d'une fonction f suffisamment difficile, alors il est possible d'appliquer f plusieurs fois pour obtenir une sortie longue qui reste pseudo-aléatoire.

***Théorème 5.3* ([NW94]).** — *S'il existe une fonction $f : \{0,1\}^t \to \{0,1\}$ qui est suffisamment difficile mais calculable en temps $2^{O(t)}$, alors pour toute constante $c > 0$, il existe un $(O(\log n), n^c, 1/10)$-PRG.*

Esquisse de démonstration. — *Construction.* On reprend l'intuition de départ, qui montre que la fonction f elle-même donne lieu à un PRG simplement en prenant $x \mapsto (x, f(x))$. On peut alors la répéter « bêtement » sur k entrées indépendantes et obtenir un PRG, qui comporte k fois plus d'erreur et qui étend kt bits à $k(t+1)$ bits. Au lieu de faire des choix indépendants, le PRG de Nisan-Wigderson fait des choix « presque » indépendants.

L'idée principale est celle d'un dessin combinatoire :

Définition 5.4. — Un (t, m, a)-*dessin combinatoire* sur $[s]$ est une famille d'ensembles $S_1, \ldots, S_m \subseteq [s]$ telle que pour tout $i \neq j$, $|S_i| = t$ et $|S_i \cap S_j| \leq a$.

Nisan et Wigderson montrent qu'il est possible de construire un dessin combinatoire en temps polynomial. Le PRG de Nisan-Wigderson effectue alors les étapes suivantes :

1. Construire un $(t, n^c, c\log n)$-dessin combinatoire $(S_1, \ldots, S_{n^c})$ sur $[s]$, pour $s = O(c\log n)$ et t approprié.
2. Soit x la graine donnée au PRG. Soit x_{S_i} la sous-chaîne de bits de x qui correspondent aux coordonnées données par S_i. Calculer $b_i = f(x_{S_i})$ pour tout $i \in [n^c]$. Sortir $b_1, \ldots, b_{n^c}$.

Analyse. Le PRG est efficace parce qu'on peut construire le dessin combinatoire en temp polynomial. Comme $|x_{S_i}| = t = O(c\log n)$ et f est calculable en temps $2^{O(t)}$, alors tous les $f(x_{S_i})$ peuvent être calculés en temps $n^{O(c)}$.

La preuve que cette construction est pseudo-aléatoire se fait par l'absurde. S'il existait un algorithme A qui distinguait la sortie du PRG de la distribution uniforme, on pourrait en déduire la construction d' un algorithme A' qui calcule f avec une précision qui contredirait la difficulté de f. Il y a alors deux idées principales : pour tout $i \neq j$, les entrées x_{S_i}, x_{S_j} sont indépendantes, sauf pour les $\log m$ bits qui correspondent à $S_i \cap S_j$; comme $\log m$ est suffisamment petit, on peut faire une sorte de recherche exhaustive qui passe en revue tous les cas possibles. ■

6. Les codes correcteurs d'erreurs

Un code correcteur d'erreur permet de corriger des erreurs sur certains bits d'un message, quand ils ont été entachés d'erreurs. Un code est un ensemble de mots, et chaque mot représente un message. On suppose qu'il y a une distance suffisamment grande entre les mots du code, pour que, même si certains bits du mot sont erronnés, le destinataire qui reçoit le mot perturbé puisse déterminer le mot du code le plus proche du mot perturbé, et ainsi corriger le message (on dit aussi le « décoder »).

6.1. Définition et existence. — Dans la suite, on identifie $\{0,1\}^n$ avec l'espace vectoriel $GF(2)^n$ de la façon naturelle. Pour $x, y \in \{0,1\}^n$ on définit la distance de Hamming $d_H(x,y) = |\{i \in [n] \mid x_i \neq y_i\}|$.

***Définition 6.1.* —** Un sous-espace vectoriel $\mathcal{C} \subseteq \{0,1\}^n$ est un (δ, ρ)-code correcteur d'erreurs linéaire binaire si :

1. La distance $\mathsf{dist}(\mathcal{C}) = \min_{x \neq y \in \mathcal{C}} d_H(x,y)$ satisfait $\mathsf{dist}(\mathcal{C}) \geq \delta n$.
2. Le taux $\mathsf{rate}(\mathcal{C}) = \dim(\mathcal{C})/n$ satisfait $\mathsf{rate}(\mathcal{C}) \geq \rho$.

Notons l'existence de la borne triviale $\delta \leq 1/2$. Le théorème suivant, dû à Gilbert et Varshamov, et prouvé en utilisant la méthode probabiliste, montre l'existence des codes avec des bons paramètres quand δ tend vers $1/2$. Ces paramètres donnent lieu à ce qu'on appelle la borne de Gilbert-Varshamov, qui relie distance et taux.

***Théorème 6.2* ([Gil52, Var57]). —** *Pour tout $\varepsilon > 0$ et tout n, il existe un $((1/2) - \varepsilon, \varepsilon^2/10)$-code correcteur d'erreurs linéaire dans l'espace $\{0,1\}^n$.*

On choisit un nombre de vecteurs $k = (1/10)\varepsilon^2 n$ vecteurs dans $\{0,1\}^n$, au hasard, et on montre que, avec forte probabilité, l'espace engendré par ces vecteurs est de dimension k et ne contient pas de vecteurs qui sont trop proches les uns des autres.

6.2. Construction explicite. — Il existe une diversité de constructions explicites de codes correcteurs d'erreurs linéaires et binaires. Une construction de code correcteurs d'erreurs est explicite s'il existe un algorithme qui, pour tout n, construise une base du code dans l'espace $\{0,1\}^n$, en temps polynomial en n. Les techniques utilisées pour ces constructions relèvent autant de l'algèbre que de la combinatoire. Chacune de ces constructions a des avantages et des inconvénients. Pour certaines, l'algorithme de décodage est très efficace. Malheureusement, à l'heure actuelle, on ne connaît aucun code explicite qui atteint la borne de Gilbert-Varshamov. Nous invitons le lecteur à consulter l'article [**Sud03**] pour une introduction à ce domaine riche et varié.

Nous décrivons ici une construction algébrique, particulièrement élégante, due à Alon *et al.*[**AGHP92**].

Théorème 6.3 **([AGHP92]).** — *Pour tout $\varepsilon > 0$, et pour tout n de la forme $n = 2^{2m}$ pour un entier m, il existe une famille de $(\frac{1}{2} - \varepsilon, \frac{2\varepsilon}{3\sqrt{n}})$-code correcteur d'erreurs linéaire, binaire, et explicite sur l'espace $\{0,1\}^n$.*

Démonstration. — On identifie le corps $GF(2^m)$ avec l'espace vectoriel $GF(2)^m$ de la façon naturelle. Puisque, par ailleurs, $n = 2^{2m}$, on peut identifier chaque élément de $[n]$ avec une paire $(x,y) \in GF(2^m)^2$. On définit alors la base $v_1, \ldots, v_k$ avec $k = \frac{2\varepsilon}{3}\sqrt{n} + 1$ comme suit : la composante $(v_i)_{x,y}$ du vecteur v_i est égale à $(x^{i-1} \mid y)$ où l'exponentiation se fait dans le corps $GF(2^m)$ et le produit scalaire se fait dans $GF(2)^m$.

C'est le choix de k qui permet de montrer la borne pour le taux du code. Pour les bornes sur la distance et la dimension, on observe d'abord que, puisque le code est linéaire, la distance du code est égale au minimum de $d_H(w,0)$ pour tout mot w dans le code.

À tout k-uplet $a_1, \ldots, a_k \in \{0,1\}$ formé de a_i non tous nuls quelconques, on associe $w = \sum_{i=1}^k a_i v_i$, et on veut vérifier que $d_H(w,0) \geq (\frac{1}{2} - \varepsilon)n$; ceci impliquera aussi que la dimension du code est k. Pour cela, associons le polynôme formel à une indéterminée $p(t) = \sum_{i=1}^k a_i t^{i-1}$, qui a $k-1$ racines au maximum.

Définissons les ensembles de coordonnées $S_x = \{(x,y) \mid y \in GF(2^m)\}$, et associons à chaque S_x le vecteur w_{S_x}, dont toute composante dans S_x vaut w, et dont toutes les autres composantes sont nulles. Si $x \in GF(2^m)$ vérifie $p(x) \neq 0$ dans le corps $GF(2^m)$, on affirme que $d_H(w_{S_x}, 0) = \frac{1}{2}|S_x|$. Ceci résulte de l'égalité

$$w_{x,y} = \sum_{i=1}^{k} a_i (x^i \mid y) = (p(x) \mid y).$$

Comme $p(x) \neq 0$, la moitié des y vérifie $(p(x) \mid y) = 1$ et l'autre moitié vérifie $(p(x) \mid y) = 0$.

En divisant les coordonnées du w en deux parties, la première regroupant les (x,y) pour lesquels $p(x) \neq 0$ et la seconde (de cardinalité au plus $k-1$) ceux pour lesquels

$p(x) = 0$ on obtient l'inégalité

$$d_H(w, 0) \geq \frac{(n - (k-1)2^m)}{2} - (k-1)2^m$$

Le choix de k implique alors l'inégalité $d_H(w, 0) \geq (\frac{1}{2} - \varepsilon)n$. ■

7. Le principe de transfert

Le principe de transfert est un outil puissant qui permet de montrer que tout sous-ensemble S d'un ensemble pseudo-aléatoire T est aussi pseudo-aléatoire. Il a été prouvé d'abord dans l'article de Green et Tao [**GT08**], puis développé dans plusieurs autres articles [**TZ08, RTTV08a, TTV09, Gow10**].

Notre présentation suit l'exposé de [**RTTV08a**]. Pour énoncer le principe formellement, nous travaillons plutôt avec les mesures au lieu des ensembles. On définit l'espérance d'une fonction f à valeur réelle sur un univers fini $\mathcal{U}$ comme

$$\mathbb{E}[f] = \tfrac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} f(u).$$

Une mesure f sur $\mathcal{U}$ est une fonction $f : \mathcal{U} \to [0, \infty[$ non-négative telle que $\mathbb{E}[f] \leq 1$. On dit qu'une mesure f est δ-dense si $\mathbb{E}[f] \geq \delta$. L'inégalité $f \leq g$ désigne que $\forall u \in \mathcal{U}, f(u) \leq g(u)$. La mesure uniforme qui vaut 1 partout est notée $\overline{1}$.

Pour tout couple de fonctions (f, g), on définit

$$(f \mid g) = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} f(u)g(u).$$

Pour un ensemble de tests $\mathcal{F} = \{F\}$ où chaque $F : \mathcal{U} \to [-1, 1]$, on dit que f et g sont ε-indistinguables par $\mathcal{F}$ si

$$\max_{F \in \mathcal{F}} |(f - g \mid F)| \leq \varepsilon$$

On dit qu'une mesure f sur $\mathcal{U}$ est ε-pseudo-aléatoire par rapport à $\mathcal{F}$ si elle est ε-indistinguable de $\overline{1}$ par $\mathcal{F}$.

Le principe de transfert affirme que tout sous-ensemble d'un ensemble pseudo-aléatoire est aussi pseudo-aléatoire. Dans le langage des mesures que nous venons de définir, cela se traduit comme suit : pour toute mesure pseudo-aléatoire ν, et pour toute $f \leq \nu$, il existe $g \leq \overline{1}$ de la même densité, indistinguable de f. Plus formellement :

Théorème 7.1 **([GT08, TZ08, RTTV08a, Gow10]).** — *Soit $\mathcal{U}$ un univers fini, $\mathcal{F}$ un ensemble de tests sur $\mathcal{U}$. Il existe alors un ensemble de tests Φ qui vérifie ce qui suit : Si ν est une mesure sur $\mathcal{U}$ qui est ε'-pseudo-aléatoire par rapport à Φ, alors, pour toute $f \leq \nu$ qui est δ-dense, il existe $g \leq \overline{1}$ qui est δ-dense, et qui est ε-indistinguable de f par $\mathcal{F}$, avec ε et ε' liés par l'égalité $\varepsilon' = \varepsilon^{O(1)}$.*

Esquisse de démonstration. — La preuve est par l'absurde : si f était distinguable de toute mesure $g \leq \overline{1}$ de densité δ, alors ν serait distinguable de $\overline{1}$.

Dans la première étape, on change l'ordre de quantificateurs, et on passe de tests dans $\mathcal{F}$ à des tests dans l'ensemble $\overline{\mathcal{F}}$ des tests qui sont des combinaisons convexes de tests dans $\mathcal{F}$.

Lemme 7.2. — *Si, pour toute $g \leq \overline{1}$ telle que $\mathbb{E}[g] = \delta$, f et g ne sont pas ε-indistinguables par $\mathcal{F}$, alors il existe $F' \in \overline{\mathcal{F}}$ telle que pour toute mesure $g \leq \overline{1}$ telle que $\mathbb{E}[g] = \delta$, $(f - g \mid F') \geq \varepsilon$.*

Le lemme est une conséquence de la dualité de la programmation linéaire, ou, d'un autre point de vue, une conséquence du théorème de Hahn-Banach.

Le Lemme 7.2 construit un distingueur universel qui distingue f de g pour toute $g \leq \overline{1}$ et $\mathbb{E}[g] = \delta$. Si on se permet un ensemble de tests Φ qui est un peu plus puissant que $\overline{\mathcal{F}}$, on peut trouver un distingueur universel $F'' \in \Phi$ *non-négatif* qui vérifie, pour toute $g \leq \overline{1}$ avec $\mathbb{E}[g] = \delta$, l'inégalité $(f - g \mid F'') \geq \varepsilon'$, où $\varepsilon' = \varepsilon^{O(1)}$.

Pour une mesure $g \leq \overline{1}$ avec $\mathbb{E}[g] = \delta$, on écrit :

$$(\nu \mid F'') = (\nu - f \mid F'') + (f \mid F'') \tag{7.1}$$
$$\geq (\nu - f \mid F'') + (g \mid F'') + \varepsilon' \tag{7.2}$$
$$= (\overline{1} \mid F'') + \varepsilon' + (\nu - f \mid F'') - (\overline{1} - g \mid F'') \tag{7.3}$$

On vérifie l'inégalité $(\nu - f \mid F'') \geq 0$, car $f \leq \nu$ et F'' non-négatif. Puis on montre l'égalité $(\overline{1} - g \mid F'') = 0$, pour g bien choisie. On déduit ainsi que $F'' \in \Phi$ distingue ν de $\overline{1}$, ce qui constitue une contradiction. ∎

7.1. Application : le théorème de Green-Tao. — Le théorème de Green-Tao affirme l'assertion suivante :

***Théorème 7.3* ([GT08]).** — *L'ensemble des nombres premiers contient des suites arithmétiques arbitrairement longues.*

La démonstration de ce théorème utilise, comme ingrédient crucial, le principe transfert. Le point de départ de la preuve de Green et Tao est le théorème de Szemerédi :

***Théorème 7.4* ([Sze75], voir aussi [Gow01]).** — *Pour tout entier positif k et tout $\delta \in]0, 1]$, il existe $N_{k,\delta}$ tel que, pour tout $N \geq N_{\delta,k}$ et tout $S \subseteq [N]$ de densité $|S| \geq \delta N$, il existe une suite arithmétique de longueur k dans S.*

Si les nombres premiers avaient une densité non-nulle dans les entiers, on pourrait alors appliquer directement le théorème de Szemerédi pour conclure qu'ils contiennent des suites arithmétiques arbitrairement longues. Malheureusement ce n'est pas le cas, car on sait que les nombres premiers ont une densité en $\Theta(1/\log N)$ parmi les nombres inférieur à N, et ainsi leur densité dans les entiers est 0.

Green et Tao contournent cet obstacle ainsi : ils ré-écrivent le théorème de Szemerédi dans le langage des mesures, utilisent le principe de transfert pour généraliser le théorème de Szemerédi : toute mesure dense majorée par une mesure pseudo-aléatoire contient des suites arithmétiques arbitrairement longues. Ensuite ils construisent une mesure dense qui est nulle partout sauf sur un sous-ensemble des nombres premiers,

et qui, en même temps, est majorée par une mesure pseudo-aléatoire. Ils obtiennent ainsi la preuve cherchée.

Ici nous esquissons la généralisation du théorème de Szemerédi et nous invitons le lecteur à consulter l'article original [**GT08**] ainsi que les autres articles [**Blo10, Kra06**] qui fournissent des démonstrations complètes.

7.1.1. Généralisation du théorème de Szemerédi. — Green et Tao montrent le résultat suivant :

***Théorème 7.5* ([GT08]).** — *Soit k un entier positif et $\delta \in]0,1]$. Il existe $N_{k,\delta}$ tel que, pour tout $N \geq N_{k,\delta}$, il existe un ensemble de tests Φ sur l'univers $\mathcal{U} = \mathbb{Z}/N\mathbb{Z}$ qui vérifie ce qui suit :*
Pour toute mesure ν sur $\mathcal{U}$ qui est $o(1)$-pseudo-aléatoire par rapport à Φ, pour toute $f \leq \nu$, il existe une suite arithmétique σ dans $\mathbb{Z}/N\mathbb{Z}$ pour laquelle on a $\prod_{i\in\sigma} f(i) > 0$.

La démonstration du théorème passe par le principe de transfert. Les auteurs donnent une définition explicite de l'ensemble Φ et ils démontrent le cas particulier suivant du principe de transfert :

Théorème 7.6. — *Soit ν une mesure sur $\mathbb{Z}/N\mathbb{Z}$ qui est ε-pseudo-aléatoire par rapport à Φ. Soit $f \leq \nu$ qui est δ-dense. Alors il existe $g \leq \overline{1}$ qui est δ-dense telle que f, g soient ε'-indistinguable par la k-norme de Gower.*

Nous invitons le lecteur à consulter [**Gow98, Gow01, Blo10**] pour une définition des normes de Gower. La propriété des normes de Gower qu'on utilise est la suivante : si la k-norme de Gower de $f-g$ est petite, alors les deux assertions sont équivalentes : – il existe une suite arithmétique σ de longueur k telle que $\prod_{i\in\sigma} f(i) > 0$ – il existe une suite arithmétique τ de longueur k telle que $\prod_{i\in\tau} g(i) > 0$.

Esquissons l'argument du Théorème 7.5 : on applique le Théorème 7.6 pour obtenir $g \leq \overline{1}$ telle que $\mathbb{E}[g] = \delta$ et telle que f et g soient indistinguables par la k-norme de Gower. La version du Théorème 7.4 pour les mesures prouve alors l'existence d'une suite arithmétique σ de longueur k telle que $\prod_{i\in\sigma} g(i) > 0$. Comme la k-norme de Gower de $f-g$ est petite, on en déduit l'existence d'une telle suite pour f.

8. Conclusion

8.1. Relations entre les objets pseudo-aléatoires. — Les objets pseudo-aléatoires ont des relations très intéressantes les uns avec les autres, et les progrès dans la compréhension d'un objet entraînent souvent une meilleure compréhension d'autres objets. Nous avons vu qu'on peut construire des extracteurs d'aléa à partir des graphes expandeurs. Beaucoup d'autres liens existent : les codes correcteurs d'erreurs binaires sont équivalents à des graphes expandeurs de Cayley sur le groupe $GF(2)^n$; les codes correcteurs d'erreurs avec décodage par liste servent à construire des extracteurs d'aléa ; le PRG de Nisan-Wigderson peut être transformé en extracteur d'aléa ; les extracteurs d'aléa peuvent être transformés en graphes expandeurs, et ainsi de suite. Les articles proposés dans les conseils de lecture aideront le lecteur à mieux comprendre ces liens.

L'exposé de Vadhan [**Vad07**] unifie formellement l'étude du pseudo-aléa en donnant un schéma général. Il montre qu'il est possible de définir une notion de code suffisamment générale telle que , si on impose des conditions supplémentaires et adaptées sur le décodage, un tel code devient un graphe expandeur, un extracteur d'aléa, un générateur pseudo-aléatoire, etc.

8.2. Développements récents. — Notre compréhension du pseudo-aléa ne cesse pas de s'approfondir. Nous avons déjà vu quelques exemples de résultats récents : l'extracteur optimal du Théorème 4.4 [**GUV09**] et les graphes expandeurs explicites construits avec le produit zig-zag sans outils algébriques [**RVW02**]. Par ailleurs, on a utilisé le produit zig-zag pour prouver qu'on peut décider la connectivité d'un graphe non-orienté en espace logarithmique [**Rei08**]. Le produit zig-zag a aussi inspiré une nouvelle preuve simplifiée du théorème de PCP [**Din06**]. On sait maintenant construire sans aucune conjecture des PRG qui sont indistinguables par les polynômes [**BV07, Vio08**] et les circuits de profondeur constante [**Bra11**].

L'interaction entre les mathématiques et le pseudo-aléa est aussi devenue très active. Le principe de transfert a été généralisé [**RTTV08b, RTTV08a, Gow10**] et on a trouvé des liens étroits entre le principe de transfert et la complexité calculatoire [**TTV09**]. Des techniques et des outils de la combinatoire additive ont été adaptées pour donner de nouvelles constructions d'extracteurs d'aléa et de graphes expandeurs [**BIW06, BKS$^+$05, Bou05, Bou07, BG08**]. On a utilisé les normes de Gowers pour construire certains PRG [**BV07**] et ces normes ont aussi trouvé des applications dans d'autres aspects de la complexité calculatoire [**ST06, Sam07**].

8.3. Questions ouvertes. — Nous rassemblons ici quelques questions intéressantes dans l'étude du pseudo-aléa qui restent ouvertes.

1. Est-il possible de construire des graphes de Ramanujan explicites, avec un rapport optimal entre le degré et l'expansion, uniquement avec des techniques combinatoires ?
2. Existe-t-il des codes correcteurs d'erreurs binaires explicites qui atteignent la borne de Gilbert-Varshamov ?
3. Existe-t-il une caractérisation algébrique des extracteurs d'aléa ?
4. Peut-on construire des PRG qui « trompent » des algorithmes restreints sans utiliser aucune conjecture ? Par exemple, des algorithmes calculables avec un circuit de profondeur constante et doté de portes qui calculent la majorité ?

8.4. Conseils de lecture. — Pour des textes introductifs, nous conseillons l'exposé de Hoory *et al.* [**HLW06**] pour les graphes expandeurs ; les exposés de Shaltiel [**Sha04**] , de Nisan et Ta-Shma [**NTS98**] pour les extracteurs d'aléa ; les exposés de Impagliazzo [**Imp02**], de Kabanets [**Kab02**], et de Trevisan [**Tre02**], pour les PRG ; les exposés de Trevisan [**Tre02**] et de Vadhan [**Vad07**] pour le pseudo-aléa en général.

Pour des textes plus détaillés, nous recommandons également le texte de Hoory *et al.* [**HLW06**] pour les graphes expandeurs ; le polycopié de cours de Wigderson [**Wig98**] pour les PRG ; le polycopié de cours de Sudan [**Sud03**] pour une introduction algorithmique à la théorie des codes correcteurs d'erreurs ; le prochain livre de Vadhan

(brouillon disponible sur Internet [**Vad10**], voir aussi son polycopié [**Vad**]) pour un traitement du pseudo-aléa, y compris des sujets que nous n'avons pas du tout traité comme la technique des espérances conditionnelles et l'indépendance par paires.

Pour un point de vue du pseudo-aléa cryptographique, nous conseillons le livre de Goldreich [**Gol00, Gol04**].

Références

[AC88] N. Alon & F. R. K. Chung – « Explicit construction of linear sized tolerant networks », *Discrete Math.* **72** (1988), p. 15–19.

[AGHP92] N. Alon, O. Goldreich, J. Håstad & R. Peralta – « Simple constructions of almost k-wise independent random variables », *Random Structures and Algorithms* **3** (1992), no. 3, p. 289–304.

[AKS02] M. Agrawal, N. Kayal & N. Saxena – « PRIMES is in P », *Ann. of Math* **2** (2002), p. 781–793.

[BG08] J. Bourgain & A. Gamburd – « Uniform expansion bounds for Cayley graphs of $SL_2(F_p)$ », *Annals of Mathematics* **167** (2008), p. 625–642.

[BIW06] B. Barak, R. Impagliazzo & A. Wigderson – « Extracting randomness using few independent sources », *SIAM Journal on Computing* **36** (2006), no. 4, p. 1095–1118.

[BKS+05] B. Barak, G. Kindler, R. Shaltiel, B. Sudakov & A. Wigderson – « Simulating independence : new constructions of condensers, ramsey graphs, dispersers, and extractors », in *Proc. 37th STOC*, ACM, 2005, p. 1–10.

[Blo10] T. Bloom – « The Green-Tao Theorem on arithmetic progressions within the primes », 2010, Dissertation. Available at `http://www.maths.bris.ac.uk/~matfb/dissertation.pdf`.

[Bou05] J. Bourgain – « More on the sum-product phenomenon in prime fields and its application », *International Journal of Number Theory* **1** (2005), p. 1–32.

[Bou07] ———, « On the construction of affine extractors », *Geometric And Functional Analysis* **17** (2007), p. 33–57, 10.1007/s00039-007-0593-z.

[Bra11] M. Braverman – « Poly-logarithmic independence fools bounded-depth boolean circuits », *Commun. ACM* **54** (2011), no. 4, p. 108–115.

[BRSW06] B. Barak, A. Rao, R. Shaltiel & A. Wigderson – « 2-source dispersers for sub-polynomial entropy and Ramsey graphs beating the Frankl–Wilson construction », in *Proc. STOC '06*, 2006, p. 671–680.

[BV07] A. Bogdanov & E. Viola – « Pseudorandom bits for polynomials », in *Proc. 48th FOCS*, 2007, p. 41–51.

[CW89] A. Cohen & A. Wigderson – « Dispersers, deterministic amplification, and weak random sources », in *Proc. 30th FOCS*, IEEE, 1989, p. 14–19.

[Din06] I. Dinur – « The PCP theorem by gap amplification », in *Proc. 38th ACM Symp. on Theory of Computing*, 2006, p. 241–250.

[Erd47] P. Erdős – « Some remarks on the theory of graphs », *Bull. Amer. Math. Soc.* **53** (1947), p. 292–294.

[FW81] P. Frankl & R. M. Wilson – « Intersection theorems with geometric consequences », *Combinatorica* **1** (1981), no. 4, p. 357–368.

[GG81] O. Gabber & Z. Galil – « Explicit constructions of linear-sized superconcentrators », *J. of Comp. and Sys. Sci.* **22** (1981), no. 3, p. 407–420.

[Gil52] E. N. Gilbert – « A comparison of signalling alphabets », *Bell Sys. Tech. J.* **31** (1952), p. 504–522.

[Gil93] D. Gillman – « A chernoff bound for random walks on expander graphs », in *Proc. FOCS '93*, 1993, p. 680–691.

[Gol00] O. Goldreich – *Foundations of Cryptography : Basic Tools*, Cambridge University Press, New York, NY, USA, 2000.

[Gol04] ———, *Foundations of Cryptography : Volume 2, Basic Applications*, Cambridge University Press, New York, NY, USA, 2004.

[Gow98] W. T. Gowers – « A new proof of Szemerédi's theorem for arithmetic progressions of length four », *Geom. Func. Anal.* **8** (1998), p. 529–551.

[Gow01] ———, « A new proof of Szemerédi's theorem », *Geom. Func. Anal.* **11** (2001), p. 465–558.

[Gow10] ———, « Decompositions, approximate structure, transference, and the Hahn-Banach theorem », *Bull. London Math. Soc.* **42** (2010), no. 4, p. 573–606.

[GT08] B. Green & T. Tao – « The primes contain arbitrarily long arithmetic progressions », *Annals of Mathematics* **167** (2008), no. 2, p. 481–547.

[GUV09] V. Guruswami, C. Umans & S. Vadhan – « Unbalanced expanders and randomness extractors from Parvaresh–Vardy codes », *J. ACM* **56** (2009), p. 20 :1–20 :34.

[Hea06] A. Healy – « Randomness-efficient sampling within NC1 », *Proc. 10th RANDOM* (2006), p. 398–409.

[HLW06] S. Hoory, N. Linial & A. Wigderson – « Expander graphs and their applications », *Bull. Amer. Math. Soc.* **43** (2006), p. 439–561.

[Imp02] R. Impagliazzo – « Hardness as randomness : a survey of universal derandomization », in *Proc. ICM*, vol. 3, 2002, p. 659–672.

[JM87] S. Jimbo & A. Maruoka – « Expanders obtained from affine transformations », *Combinatorica* **7** (1987), no. 4, p. 343–355.

[Kab02] V. Kabanets – « Derandomization : a brief overview », *Bulletin of the EATCS* **76** (2002), p. 88–103.

[KI03] V. Kabanets & R. Impagliazzo – « Derandomizing polynomial identity tests means proving circuit lower bounds », in *Proc. 35th STOC*, ACM, 2003, p. 355–364.

[Kra06] B. Kra – « The Green-Tao Theorem on arithmetic progressions in the primes : an ergodic point of view », *Bull. Amer. Math. Soc.* **43** (2006), p. 3–23.

[LPS88] A. Lubotzky, R. Phillips & P. Sarnak – « Ramanujan graphs », *Combinatorica* **8** (1988), no. 3, p. 261–277.

[Mar88] G. A. Margulis – « Explicit group-theoretic constructions of combinatorial schemes and their applications in the construction of expanders and concentrators », *Problemy Peredachi Informatsii* **24** (1988), no. 1, p. 51–60.

[Mil75] G. L. Miller – « Riemann's hypothesis and tests for primality », in *Proc. 7'th STOC*, ACM, 1975, p. 234–239.

[Nil91] A. Nilli – « On the second eigenvalue of a graph », *Discrete Mathematics* **91** (1991), no. 2, p. 207 – 210.

[NTS98] N. Nisan & A. Ta-Shma – « Extracting randomness : A survey and new constructions », *Journal of Computer and System Sciences* **58** (1998), p. 148–173.

[NW94] N. NISAN & A. WIGDERSON – « Hardness vs randomness », *J. of Comp. and Sys. Sci.* **49** (1994), no. 2, p. 149–167, Preliminary version in FOCS' 88.

[Rab80] M. O. RABIN – « Probabilistic algorithm for testing primality », *J. Number Theory* **12** (1980), no. 1, p. 128–138.

[Ram30] F. P. RAMSEY – « On a problem of formal logic », *Proceedings of The London Mathematical Society* **s2-30** (1930), p. 264–286.

[Rei08] O. REINGOLD – « Undirected connectivity in log-space », *J. ACM* **55** (2008), p. 17 :1–17 :24.

[RTS00] J. RADHAKRISHNAN & A. TA-SHMA – « Bounds for dispersers, extractors, and depth-two superconcentrators », *SIAM J. Discret. Math.* **13** (2000), p. 2–24.

[RTTV08a] O. REINGOLD, L. TREVISAN, M. TULSIANI & S. VADHAN – « New Proofs of the Green-Tao-Ziegler Dense Model Theorem : An Exposition », *ArXiv e-prints* (2008), no. 0806.0381.

[RTTV08b] O. REINGOLD, L. TREVISAN, M. TULSIANI & S. VADHAN – « Dense subsets of pseudorandom sets », in *Proc. 49th FOCS*, 2008, p. 76–85.

[RVW02] O. REINGOLD, S. VADHAN & A. WIGDERSON – « Entropy waves, the zig-zag graph product, and new constant-degree expanders and extractors », *Annals of Mathematics* **155** (2002), no. 1, p. 157–187.

[Sam07] A. SAMORODNITSKY – « Low-degree tests at large distances », in *Proc. 39th STOC*, 2007, p. 506–515.

[Sch80] J. T. SCHWARTZ – « Fast probabilistic algorithms for verification of polynomial identities », *J. ACM* **27** (1980), p. 701–717.

[Sha04] R. SHALTIEL – « Recent developments in extractors », in *Current Trends in Theoretical Computer Science : The Challenge of the New Century* (G. Paun, G. Rozenberg & A. Salomaa, éds.), World Scientific Press, 2004, p. 189–228.

[SS77] R. SOLOVAY & V. STRASSEN – « A fast monte-carlo test for primality », *SIAM Journal on Computing* **6** (1977), no. 1, p. 84–85.

[ST06] A. SAMORODNITSKY & L. TREVISAN – « Gowers uniformity, influence of variables, and PCPs », in *Proc. 38th STOC* (New York, NY, USA), STOC '06, ACM, 2006, p. 11–20.

[Sud03] M. SUDAN – « Essential coding theory », 2003, Lecture notes. Available from `http://people.csail.mit.edu/madhu/FT04/`.

[Sze75] E. SZEMERÉDI – « On sets of integers containing no k elements in arithmetic progression », *Acta Mathematica* **27** (1975), p. 199–245.

[Tre02] L. TREVISAN – « Pseudorandomness and combinatorial constructions », in *Proc. ICM*, vol. 3, 2002, p. 1111–1136.

[TTV09] L. TREVISAN, M. TULSIANI & S. VADHAN – « Regularity, boosting, and efficiently simulating every high-entropy distribution », in *Proc. 24th CCC*, CCC '09, 2009, p. 126–136.

[TZ08] T. TAO & T. ZIEGLER – « The primes contain arbitrarily long polynomial progressions », *Acta Mathematica* **201** (2008), p. 213–305.

[Vad] S. VADHAN – « Pseudorandomness », Lectures notes. Available at `http://people.seas.harvard.edu/~salil/cs225/`.

[Vad07] S. VADHAN – « The unified theory of pseudorandomness », *SIGACT News* **38** (2007), p. 39–54.

[Vad10] S. VADHAN – « Pseudorandomness », 2010, Monograph. Draft available at `http://people.seas.harvard.edu/~salil/pseudorandomness/pseudorandomness-Aug10.pdf`.

[Var57] R. R. Varshamov – « Estimate of the number of signals in error correcting codes », *Dokl. Acad. Nauk SSSR* **117** (1957), p. 739–741.

[Vio08] E. Viola – « The sum of d small-bias generators fools polynomials of degree d », in *Proc. CCC*, 2008, p. 124–127.

[Wig98] A. Wigderson – « Derandomizing BPP », 1998, Lecture notes. Available at `http://www.math.ias.edu/~avi/BOOKS/rand.pdf`.

[Xia03] D. Xiao – « The evolution of expander graphs », 2003, AB thesis, Harvard University.

[Zip79] R. Zippel – « Probabilistic algorithms for sparse polynomials », in *Proc. Intl. Sym. on Symbolic and Algebraic Computation* (London, UK), Springer-Verlag, 1979, p. 216–226.

[Zuc07] D. Zuckerman – « Linear degree extractors and the inapproximability of max clique and chromatic number », *Theory of Computing* **3** (2007), no. 1, p. 103–128.

D. Xiao, LIAFA, Université Paris Didérot - Paris 7, Case 7014, 75205 Paris Cedex 13
E-mail : `dxiao@liafa.jussieu.fr`