Fabriquer du hasard avec des conjectures d'informatique théorique

Nicolas Schabanel
CNRS, Université Paris Diderot (LIAFA)
École ALÉA 2010

Le hasard existe bien... mais comment en engendrer ? comment le mesurer ?

Solomonoff (1965), Kolmogorov, Chaitin (1966)

- K-complexité(*w*) = la taille du plus petit programme engendrant le mot binaire *w*

Définition déterministe du hasard

- Les mots *incompressibles,* de K-complexité maximale (> 0.99|w|), vérifient les mêmes propriétés statistiques que les suites aléatoires

Problème: non-calculable !

Poincaré (1890), Hadamard (1898), Lorenz (1960-)

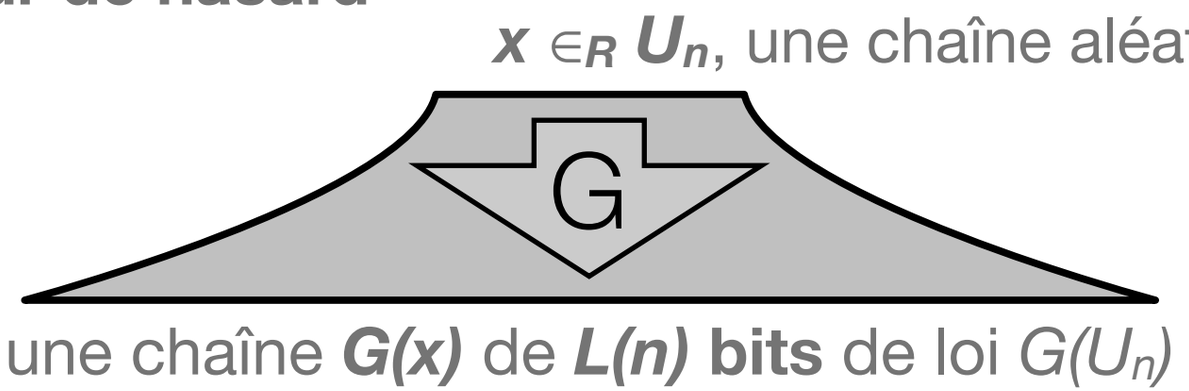
- Découverte des systèmes dynamiques chaotiques

« Une cause très petite, qui nous échappe, détermine un effet considérable que nous ne pouvons pas ne pas voir, et alors nous disons que cet effet est dû au hasard. »

- ***Problème: est-ce vraiment du vrai hasard ?***

Cryptographie: Engendrer du suffisamment de hasard pour un utilisateur de capacité limitée

- **Générateur de hasard**



- **Utilisateur** = un **algorithme *A* en temps polynomial** qui cherche à déterminer si la distribution est aléatoire ou non

- ***G*** est un **générateur *pseudo-aléatoire*** si pour tout algorithme randomisé ***A* en temps polynomial,**

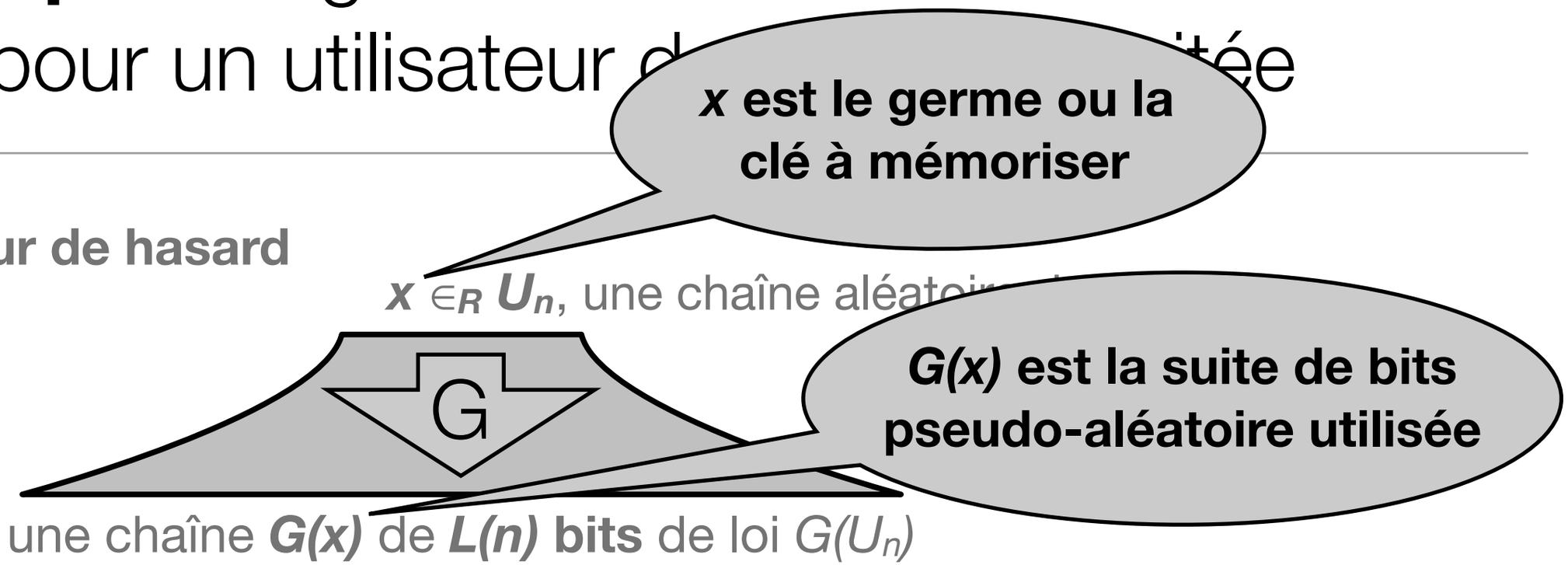
$$\left| \Pr_{x \in_R U_n} \{A(G(x)) = 1\} - \Pr_{y \in_R U_{L(n)}} \{A(y) = 1\} \right| < \varepsilon(n)$$

où $\varepsilon(n) = o(n^{-c})$ pour tout c > 0.

A ne peut distinguer de manière significative $G(U_n)$ de $U_{L(n)}$

Cryptographie: Engendrer du suffisamment de hasard pour un utilisateur d[illegible]ité

- **Générateur de hasard**



- **Utilisateur** = un **algorithme *A* en temps polynomial** qui cherche à déterminer si la distribution est aléatoire ou non

- ***G*** est un **générateur *pseudo-aléatoire*** si pour tout algorithme randomisé ***A* *en temps polynomial,***

$$\left| \Pr_{x \in_R U_n} \{A(G(x)) = 1\} - \Pr_{y \in_R U_{L(n)}} \{A(y) = 1\} \right| < \varepsilon(n)$$

où $\varepsilon(n) = o(n^{-c})$ pour tout c > 0.

A ne peut distinguer de manière significative $G(U_n)$ de $U_{L(n)}$

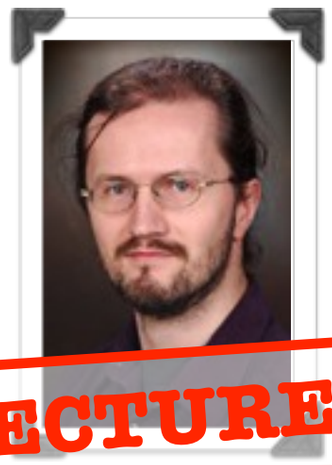
Cryptographie: Engendrer du suffisamment de hasard pour un utilisateur de capacité limitée

GÉNÉRATEUR PSEUDO-ALÉATOIRE

GÉNÉRATEUR IMPRÉVISIBLE

EXISTENCE D'UNE FONCTION

***RÉELLEMENT* DURE À CALCULER (OU À INVERSER)**



≈P/POLY=?EXP, CONJECTURE SIMILAIRE À P=?NP

Générateurs pseudo-aléatoire et imprévisible

- ***G*** est un **générateur *pseudo-aléatoire*** si pour <u>tout</u> algorithme randomisé ***A*** ***en temps polynomial,***

$$\left| \Pr_{x \in_R U_n} \{A(G(x)) = 1\} - \Pr_{y \in_R U_{L(n)}} \{A(y) = 1\} \right| < \varepsilon(n)$$

où $\varepsilon(n) = o(n^{-c})$ pour tout c > 0.

A ne peut distinguer de manière significative $G(U_n)$ de $U_{L(n)}$

- ***G*** est un **générateur imprévisible** si pour <u>tout</u> algorithme polynomial B

$$\Pr_{\substack{x \in_R U_n \\ y = G(x) \\ i \in_R \{1,\ldots,L(n)\}}} \{ B(1^n, y_1, \ldots, y_{i-1}) = y_i \} \leqslant \frac{1}{2} + \varepsilon(n)$$

***B* ne peut pas prédire un bit aléatoire de *G(x) en* fonction des précédents**
avec une probabilité significativement supérieure à 1/2

Ce que l'on va voir dans cet exposé

- **Théorème (Yao, 1982])** Tout générateur imprévisible est pseudo-aléatoire *(et réciproquement).*
- **Théorème (Goldreich, Levin, 1989)** S'il existe une permutation à sens unique, alors on peut construire un générateur imprévisible ***G*** d'extension $L(n) = n^c$ pour tout $c > 0$.

L'occasion d'apprendre un certain nombre de *techniques génériques*

- **Remplacer le futur par du hasard:** utile aussi pour la dérandomisation par la méthode de l'espérance conditionnelle
- **Hybridation:** déformation progressive d'une distribution à une autre
- **Auto-réductibilité à une instance aléatoire:** utile aussi pour l'auto-débuggage
- **Génération économe de variables aléatoires uniformes 2-à-2 indépendantes:** utile aussi pour la construction de fonction de hashage
- **Divination exhaustive:** utile aussi pour les algorithmes d'approximation

Théorème de Yao (1982)

Tout générateur imprévisible est pseudo-aléatoire

(et réciproquement).

Théorème de Yao (1982) Tout générateur imprévisible est pseudo-aléatoire *(et réciproquement)*.

Réciproque immédiate: si on peut prédire un bit aléatoire de $G(x)$, $x \in_R U_n$, en fonction des précédents, on peut distinguer $G(U_n)$ et $U_{L(n)}$

Preuve par contraposée: non pseudo-aléatoire ⇒ non imprévisible.

- **Soit *G* qui n'est pas pseudo-aléatoire:** Soit ***A*** un algorithme polynomial distinguant $G(U_n)$ de $U_{L(n)}$, i.e., tel qu'il existe c, tel que pour une infinité de ***n***

$$\left| \Pr_{x \in_R U_n} \{A(G(x)) = 1\} - \Pr_{y \in_R U_{L(n)}} \{A(y) = 1\} \right| > n^{-c}$$

- **Un algorithme pour prédire G:** Considérons l'algorithme ***B*** suivant

Entrée: 1^n, indice $i \in_R U_{L(n)}$, $y_1,\ldots,y_{i-1}$

Sortie: prédiction de la valeur de y_i

- Tirer des bits aléatoires $z_i,\ldots,z_n$ et poser $a = A(y_1,\ldots,y_{i-1},z_i,\ldots,z_n)$
- *Si* $a = 1$, alors renvoyer z_i *(le choix de* z_i *est présumé correct)*, sinon $1-z_i$

Théorème de Yao (1982) Tout générateur imprévisible est pseudo-aléatoire *(et réciproquement).*

Réciproque immédiate: si on peut prédire un bit aléatoire de $G(x)$, $x \in_R U_n$, en fonction des précédents, on peut distinguer $G(U_n)$ et $U_{L(n)}$

Preuve par contraposée: non pseudo-aléatoire ⇒ non imprévisible.

- **Soit *G* qui n'est pas pseudo-aléatoire:** Soit ***A*** un algorithme polynomial distinguant $G(U_n)$ de $U_{L(n)}$, i.e., tel qu'il existe c, tel que pour une infinité de ***n***

$$\Pr_{x \in_R U_n}\{A(G(x)) = 1\} - \Pr_{y \in_R U_{L(n)}}\{A(y) = 1\} > n^{-c}$$

- **Un algorithme pour prédire G:** Considérons l'algorithme ***B*** suivant

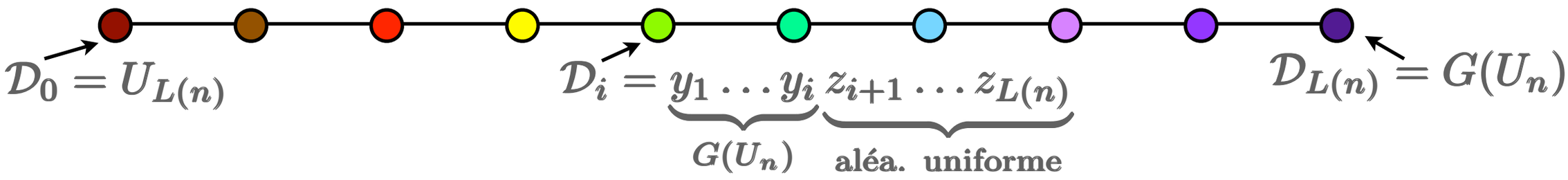
Entrée: 1^n, indice $i \in_R U_{L(n)}$, $y_1,\ldots,y_{i-1}$

Sortie: prédiction de la valeur de y_i

- Tirer des bits aléatoires $z_i,\ldots,z_n$ et poser $a = A(y_1,\ldots,y_{i-1},z_i,\ldots,z_n)$
- *Si* $a = 1$, alors renvoyer z_i *(le choix de* z_i *est présumé correct)*, sinon $1-z_i$

Hybridation

- **Fixons n, nous avons:** $\Pr_{x \in_R U_n}\{A(G(x)) = 1\} > \Pr_{y \in_R U_{L(n)}}\{A(y) = 1\} + \varepsilon \quad (\varepsilon = n^{-c})$
- **Démontrons que:** $\Pr_{\substack{x \in_R U_n \\ y = G(x) \\ i \in_R \{1,\ldots,L(n)\}}}\left\{ B(1^n, y_1, \ldots, y_{i-1}) = y_i \right\} \geqslant \frac{1}{2} + \frac{\varepsilon}{L(n)} = \frac{1}{2} + \Omega\left(\frac{1}{n^{c'}}\right)$
- **Hybridation:** On définit ***L(n)+1*** distributions



la distribution D_i consiste à choisir $x \in_R U_n$, poser $y = G(x)$, choisir $L(n)$ bits aléatoires $z_1,\ldots,z_{L(n)}$ et renvoyer $y_1 \ldots y_i z_{i+1} \ldots z_{L(n)}$

- Posons: $p_i = \Pr_{t \in_R \mathcal{D}_i}\{A(t) = 1\}$
- Nous avons : $p_{L(n)} - p_0 \geqslant \varepsilon$ c'est-à-dire : $\mathbb{E}_{i \in_R \{1,\ldots,L(n)\}}[p_i - p_{i-1}] \geqslant \frac{\varepsilon}{L(n)}$

B prédit $G(U_n)$

- **Y'a plus qu'à prouver que :** $\Pr_{\substack{x \in_R U_n \\ y = G(x)}} \{ B(1^n, y_1, \ldots, y_{i-1}) = y_i \} = \frac{1}{2} + \underbrace{p_i - p_{i-1}}_{\mathbb{E}_i[\ ] \geq \frac{\varepsilon}{L(n)}}$

Algorithme B

Entrée: 1^n, indice $i \in_R U_{L(n)}$, $y_1,\ldots,y_{i-1}$

Sortie: prédiction de la valeur de y_i

- Tirer des bits aléatoires $z_i,\ldots,z_n$ et poser $a = A(y_1,\ldots,y_{i-1},z_i,\ldots,z_n)$
- *Si* $a = 1$, alors renvoyer z_i *(le choix de* z_i *est présumé correct)*, sinon $1-z_i$

- ***B*** prédit correctement y_i si ($z_i = y_i$ et $a = 1$) ou ($z_i = 1-y_i$ et $a = 0$), i.e.

$$\frac{1}{2}\underbrace{\Pr\{a=1|z_i=y_i\}}_{p_i} + \frac{1}{2}\Big(1 - \underbrace{\Pr\{a=1|z_i=1-y_i\}}_{2\underbrace{\Pr\{a=1\}}_{p_{i-1}} - \underbrace{\Pr\{a=1|z_i=y_i\}}_{p_i}}\Big) = \frac{1}{2} + p_i - p_{i-1}$$

Fin de la première étape

- Comme B est un algorithme ***polynomial*** nous avons bien:

Théorème de Yao (1982)

Tout générateur imprévisible est pseudo-aléatoire

(et réciproquement).

Proposition (Goldreich, Levin, 1989)

Si $f : \{0,1\}^* \to \{0,1\}^*$ est une permutation à sens unique, alors $G(x,r) = \langle r, f(x), x \odot r \rangle$ est un générateur imprévisible ajoutant un bit

où $x \odot r = \sum_{i=1}^{n} x_i \cdot r_i \pmod 2$

Proposition (Goldreich, Levin, 1989)

Si $f : \{0,1\}^* \to \{0,1\}^*$ est une permutation à sens unique, alors $G(x,r) = \langle r, f(x), x \odot r \rangle$ est un générateur imprévisible ajoutant un bit

où $x \odot r = \sum_{i=1}^{n} x_i \cdot r_i \pmod 2$

forme linéaire aléatoire

Permutation à sens unique

- $f : \{0,1\}^* \to \{0,1\}^*$ est une **permutation à sens unique** si:
  - f est bijective
  - pour tout mot ***x***, $|f(x)| = |x|$
  - pour tout algorithme polynomial ***I,*** il existe **ε**, tel que pour tout ***n***

$$\Pr_{x \in_R U_n}\left\{ I(f(x)) = x \right\} < \varepsilon(n) \quad \text{où } \varepsilon(n) = o(n^{-c}) \text{ pour tout } c > 0$$

- ***NB: l'existence d'une telle fonction implique que P ≠ NP***
- **Fonctions et permutations candidates :**
  - **RSA: $x \mapsto x^e \bmod N$** pour $N = pq$ où p et q sont premiers et secrets
  - **Rabin: $x \in QR_N \mapsto x^2 \bmod N$** pour $N = pq$ où p et q sont premiers, secrets et **= 1 mod 4**

Les bons ***x*** sont assez nombreux

$$G(x,r) = \langle r, f(x), x \odot r \rangle \text{ avec } x \odot r = \sum_{i=1}^{n} x_i \cdot r_i \pmod 2$$

- **Par l'absurde:**
  il existe un algorithme polynomial ***A*** qui prédit le dernier bit, ***x*⊙*r,*** à partir de ***f(x)*** et ***r*** avec probabilité **≥ 1/2+ε** pour une infinité de ***n*** (où **ε** = n^{-c}).
  Fixons un tel ***n***.

➡ Exhibons un algorithme ***I*** qui inverse ***f*** sur une **fraction** non-négligeable des ***x*** avec une probabilité significative, c.-à-d. tel que

$$\Pr_{x \in_R U_n} \left\{ I(f(x)) = x \right\} \geqslant \Omega(\varepsilon)$$

- **Les *x* candidats :** Par un argument de comptage, il existe une **fraction ≥ ε/2 des *x*,** tels que ***A*** prédit correctement ***x*⊙*r*** à partir de ***f(x)*** et ***r*** avec probabilité **≥ 1/2+ε/2**

  De tels ***x*** sont dits **bons**, on ne considère plus que les ***x*** bons.

Auto-réductibilité à une instance aléatoire

- **Si x est *"super-bon"*, c'est-à-dire si :** $\Pr_{r \in_R U_n} \left\{ A(r, f(x)) = x \odot r \right\} \geq 0.9$
- **Auto-réductibilité :** $x_i = x \odot e^i = \underbrace{(x \odot r)}_{\text{prédit par } A \text{ à } 90\%} \oplus \underbrace{(x \odot \overbrace{(r \oplus e^i)}^{\text{aléa. unif.}})}_{\text{prédit par } A \text{ à } 90\%}$ <u>OK à 80%</u>

Algorithme *I*

<u>Entrée:</u> *F*

<u>Sortie:</u> prédiction de *x tel que f(x)=F*

- Choisir $r^1,\ldots,r^m$ vecteurs aléatoires 2-à-2 indépendants de *n* bits *(m = 200n)*
- Pour *i=1..n*
  - Calculer les valeurs: $z_1 = A(r^1, F) \quad \text{et} \quad z'_i = A(r^1 \oplus e^i, F),$ $\vdots$ $z_m = A(r^m, F) \quad \text{et} \quad z'_m = A(r^m \oplus e^i, F).$
  - Renvoyer pour x_i la valeur MAJORITE{ $z_i \oplus z'_i$ }

Auto-réductibilité à une instance aléatoire

- **Si x est *"super-bon"*, c'est-à-dire si :** $\Pr_{r \in_R U_n} \left\{ A(r, f(x)) = x \odot r \right\} \geq 0.9$
- **Auto-réductibilité :** $x_i = x \odot e^i = \underbrace{(x \odot r)}_{\text{prédit par } A \text{ à } 90\%} \oplus \underbrace{(x \odot \overbrace{(r \oplus e^i)}^{\text{aléa. unif.}})}_{\text{prédit par } A \text{ à } 90\%}$ <u>OK à 80%</u>

Comme les r^i sont 2-à-2 indépendants, la majorité donne la réponse correcte par Chebychev pour m assez grand !

<u>Entrée:</u> *F*

<u>Sortie:</u> prédiction de *x tel qu*[illegible]

- Choisir $r^1,\ldots,r^m$ vecteurs aléatoires 2-à-2 indépendants de *n* bits *(m = 200n)*
- Pour *i=1..n*
  - Calculer les valeurs: $z_1 = A(r^1, F) \quad \text{et} \quad z'_i = A(r^1 \oplus e^i, F),$ $\vdots$ $z_m = A(r^m, F) \quad \text{et} \quad z'_m = A(r^m \oplus e^i, F).$
  - Renvoyer pour x_i la valeur MAJORITE{ $z_i \oplus z'_i$ }

L’auto-réductibilité ne suffit plus !

- **Supposons maintenant que x est seulement bon:**

$$\Pr_{r \in_R U_n} \{ A(r, f(x)) = x \odot r \} \geqslant \frac{1}{2} + \frac{\varepsilon}{2}$$

- **L’auto-réductibilité ne suffit pas !**

$$x_i = x \odot e^i = \underbrace{(x \odot r)}_{\text{prédit par } A \text{ à } 50\% + \frac{\varepsilon}{2}} \oplus \underbrace{(x \odot \overbrace{(r \oplus e^i)}^{\text{aléa. unif.}})}_{\text{prédit par } A \text{ à } 50\% + \frac{\varepsilon}{2}}$$

$$\text{correct avec une probabilité } \epsilon \ll 50\% \text{ insuffisante !}$$

- **Il faut réussir à n’avoir à prendre la majorité que sur l’un des deux termes seulement ! Il va falloir deviner les valeurs de l’autre !**

Espace de vecteurs 2-à-2 indépendants

- Nous n'avons besoin que de vecteurs de n bits $r^1,\dots,r^m$ uniformes et **2-à-2 indépendants**

- Soit k tel que $2^{k-1} < m+1 \leq 2^k$ et $s^1,\dots,s^k$ vecteurs aléatoires uniformes indépendants à n bits ($k \sim \log m = O(\log n)$)

- Pour $j=1,\dots,m$ on note j_q le q-ème bit de l'écriture binaire de j sur k bits.

- **Théorème.** Les vecteurs aléatoires $r^1,\dots,r^m$ donnés par

$$r^j = \bigoplus_{q:j_q=1} s^q$$

forment une famille de vecteurs à n bits **uniformes** et **2-à-2 indépendants**.

Divination exhaustive

- On doit réussir à prédire avec succès à la fois :

  $x \odot r^j$ et $x \odot (r^j \oplus e^j)$

  pour en déduire $x_i = (x \odot r^j) \oplus (x \odot (r^j \oplus e^j))$

- **Divination exhaustive :**

  On peut chercher à **deviner** les valeurs de $x \odot r^j$ par énumération **exhaustive !**
  En effet, pour tout j,

$$x \odot r^j = \bigoplus_{q : j_q = 1} (x \odot s^q)$$

  Il suffit donc de décider des k valeurs de $x \odot s^q \in \{0,1\}$ pour obtenir les m valeurs des $x \odot r^j$ **!**
  Il y a $2^k \leq 2m$ valeurs possibles à énumérer, on peut procéder par **énumération exhaustive en temps polynomial !**

Divination exhaustive

Algorithme d'inversion *I*

Entrée: ***F***

Sortie: ***x*** tel que ***f(x) = F***

- Soit ***k*** tel que $2^{k-1} < m+1 \leq 2^k$ où $m = 20n/\varepsilon^2$
- Soient $s^1, \ldots, s^k \in_R U_n$, ***k*** vecteurs aléatoires uniformes et indépendants de ***n*** bits
- Poser $r^j = \bigoplus_{q:j_q=1} s^q \quad \text{pour } j = 1..m$ (nos ***m*** vecteurs 2-à-2 indépendants)
- Calculer $z'_{ij} = \underbrace{A(r^j \oplus e^i, F)}_{\text{la prédiction de } A \text{ pour } x \odot (r^j \oplus e^i)} \quad \text{pour } i = 1..n \text{ et } j = 1..m$
- Pour tous les ***k***-uplets $(\sigma_1, \ldots, \sigma_k) \in \{0,1\}^k$ (nos hypothèses pour les $x \odot s^q$)
  - Poser $z_j = \underbrace{\bigoplus_{q:j_q=1} \sigma_q}_{\text{notre hypothèse pour } x \odot r^j} \quad \text{pour } j = 1..m$
  - Considérons ξ avec $\xi_i = \text{MAJORITÉ}_j\{z_j \oplus z'_{ij}\} \quad \text{pour } i = 1..n$
  - Si ***f(x) = F*** alors STOP et renvoyer ***x*** = ξ sinon tester le ***k***-uplet suivant
- STOP et renvoyer ERREUR

Analyse

- Une fraction **≥ε/2** des ***x*** sont bons et vérifient : $\Pr_{r \in_R U_n}\{A(r, f(x)) = x \odot r\} \geqslant \frac{1}{2} + \frac{\varepsilon}{2}$
- Considérons un tel ***x*** et posons ***F = f(x)***. Étudions l'algorithme ***I*** sur l'entrée ***F***.
- Considérons l'itération où le **k**-uplet $(\sigma_1, \ldots, \sigma_k) = (x \odot s^1, \ldots, x \odot s^k)$ correct est testé.
  Par construction, on a bien : $z_j = x \odot r^j \quad \textbf{pour tout } 1 \leqslant j \leqslant m$
- Considérons les variables aléatoires suivantes:
  $$Z_{ij} = \mathbb{1}_{\{A(r^j \oplus e^i, F) = x \odot (r^j \oplus e^i)\}} \quad \text{et} \quad Z_i = \sum_{j=1}^{m} Z_{ij}$$
- Les bits de ***x*** sont donc correctement prédits dans **ξ** si et seulement si $Z_i >$ **m/2** pour tout ***i***
- Les Z_{ij} sont des variables aléatoires 2-à-2 indépendantes d'espérance **≥ 1/2+ε/2** et de variance **≤ 1**. La variance de chaque Z_i est donc **≤ *m***. Par Chebychev, $\Pr\{Z_i \leqslant m/2\} \leqslant 1/5n \textbf{ et donc } \Pr\{(\exists i)\ Z_i \leqslant m/2\} \leqslant 1/5$.
- On aboutit donc à la contradiction : $\Pr_{x \in_R U_n}\{I(f(x)) = x\} \geqslant \frac{\varepsilon}{10} = \Omega(n^{-c})$
  f ne peut donc être à sens unique !

Extension à n^c bits

Théorème (Goldreich, Levin, 1989)

Si $f : \{0,1\}^* \to \{0,1\}^*$ est une permutation à sens unique telle que $|f(x)| = |x|$ pour tout mot x, alors

$$G(x,r) = \langle r, f^L(x) \odot r, f^{L-1}(x) \odot r, \ldots, f(x) \odot r \rangle$$

est un générateur imprévisible ajoutant *L*-*n* bits pour tout $L \leq n^c$ avec *c* entier

En guise de conclusion:

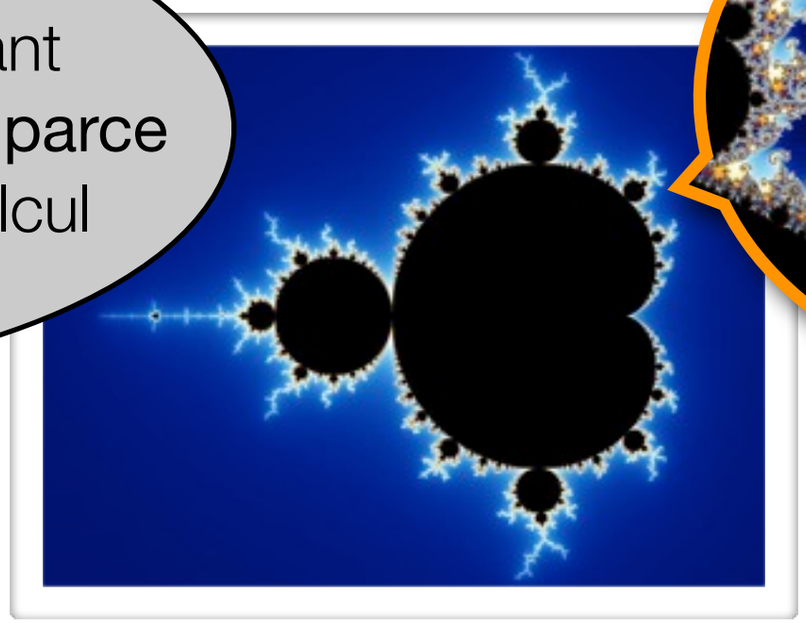
Même si le hasard n'existe pas dans la nature, et que l'on ne peut fabriquer du hasard véritable, la théorie des probabilités s'applique !

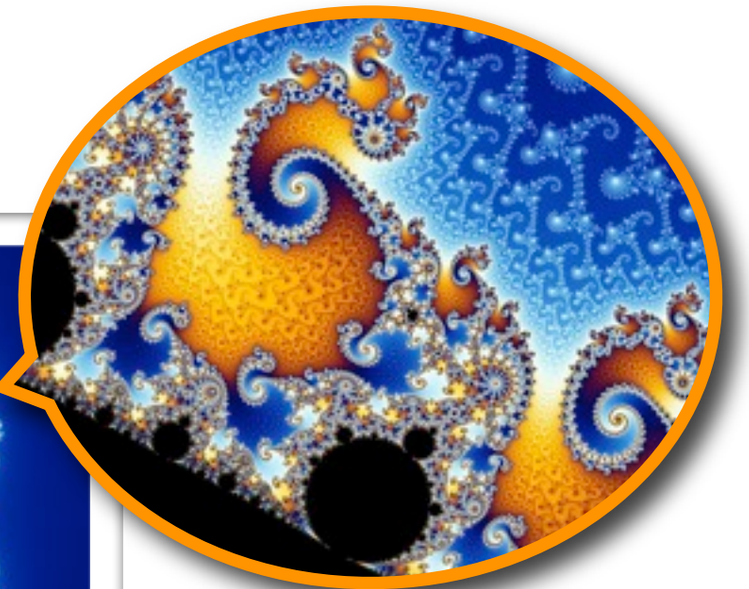
(pour peu que les conjectures d'informatique théorique soient vérifiées !)

Pourquoi ces questions m'intéressent ?

- Ces travaux définissent une notion d'aléatoire relative à la puissance d'un observateur: sorte de complexité maximale (au sens de Kolmogorov) relative à un observation polynomial
- Pourrait-on définir une notion de complexité relative à un observateur polynomial ? Quelles seraient les conjectures d'informatique qui seraient liées à l'existence d'objets "complexes à observer" (et non plus pseudo-aléatoires) ?

Puis-je qualifier de "complexe" cet objet pourtant "simple" selon la K-complexité, **parce que** j'ai une puissance de calcul **limitée** ?





Référence : Chapitre 9 du livre d'Arora et Barak