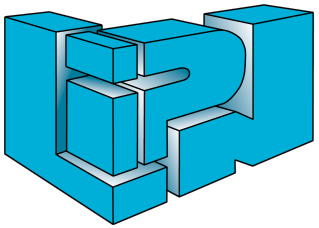
Génération aléatoire exacte de Boltzmann avec un oracle approximatif

Journées Combinatoires de Bordeaux 2014
— 12 février 2014 —

Olivier Bodini
Nicolas Rolin
Laboratoire d'Info de Paris Nord

Jérémie Lumbroso
Math Department of Simon Fraser University



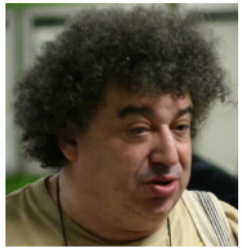
SFU

1. introduction

- Dans cet exposé : génération aléatoire d'objets combinatoires
- Extension de la méthode de Boltzmann pour "éviter" l'évaluation de séries génératrices
- **Plan**
  1. introduction
  2. générateurs de Boltzmann
  3. Boltzmann sans évaluation
  4. arbres simples
  5. arbres de Cayley
  6. conclusion

2. générateurs de Boltzmann

[Duchon, Flajolet, Louchard & Schaeffer 2001]

Soit $\mathcal{C}$ une classe, et $C(z) = \sum c_n z^n$, un générateur de Boltzmann tire un objet $\gamma \in \mathcal{C}$ avec proba

$$\mathbb{P}_z[\gamma] = \frac{z^{|\gamma|}}{C(z)}.$$

Propriétés :

- distribution de la taille : $\mathbb{P}_z[N = n] = c_n z^n / C(z)$
- uniformité à taille fixée (deux objets de même taille = même proba d'être tirés)

$$\mathbb{P}_z[\gamma \in \mathcal{C} \mid |\gamma| = n] = \frac{1}{c_n}$$

- le contrôle de la taille se fait par le choix du paramètre z

les constructeurs

(ici dans le cas étiqueté que nous ne considerons pas!)

construction	**algorithm**
$\mathcal{A} = \varepsilon$ or $\mathcal{Z}$	$\Gamma\mathcal{A}(z) :=$ **return** $\square$ or $\blacksquare$
$\mathcal{A} = \mathcal{B} + \mathcal{C}$	$\Gamma\mathcal{A}(z) :=$ **if** $\mathrm{Ber}(B(z)/(B(z)+C(z))) = 1$ **then return** $\Gamma\mathcal{B}(z)$ **else return** $\Gamma\mathcal{C}(z)$
$\mathcal{A} = \mathcal{B} \times \mathcal{C}$	$\Gamma\mathcal{A}(z) :=$ **return** $< \Gamma\mathcal{B}(z); \Gamma\mathcal{C}(z) >$
$\mathcal{A} = \mathsf{Seq}(\mathcal{B})$	$\Gamma\mathcal{A}(z) := k \leftarrow \mathrm{Geo}(A(z))$; **return** k indep. $\Gamma\mathcal{B}(z)$
$\mathcal{A} = \mathsf{Set}(\mathcal{B})$	$\Gamma\mathcal{A}(z) := k \leftarrow \mathsf{Poi}(A(z))$; **return** k indep. $\Gamma\mathcal{B}(z)$
$\mathcal{A} = \mathsf{Cyc}(\mathcal{B})$	$\Gamma\mathcal{A}(z) := k \leftarrow \mathsf{Loga}(A(z))$; **return** k indep. $\Gamma\mathcal{B}(z)$

remarques

- Génération de Boltzmann = objets plongés dans une loi en série (PSD: *power series distribution*)

- **Inconvénients :**
  - la taille est une variable aléatoire
  - il faut évaluer les séries génératrices (pb. de l'oracle)

- **Avantage :** les générateurs ont les mêmes bonnes prop. algébriques des séries génératrices (qui "permettent" la méthode symbolique)
  - élégance et simplicité des constructeurs
  - (souvent) un nombre constant de probabilités à calculer
  - indépendance des appels récursifs (parallélisation)

l'oracle [Pivoteau, Salvy, Soria 2011/2]

Algorithms for Combinatorial Structures:
Well-Founded Systems and Newton Iterations*

Carine Pivoteau† Bruno Salvy‡ Michèle Soria§

May 25, 2012

This article is dedicated to the memory of Philippe Flajolet

Abstract

We consider systems of recursively defined combinatorial structures. We give algorithms checking that these systems are well founded, computing generating series and providing numerical values. Our framework is an articulation of the constructible classes of Flajolet & Sedgewick with Joyal's species theory. We extend the implicit species theorem to structures of size zero. A quadratic iterative Newton method is shown to solve well-founded systems combinatorially. From there, truncations of the corresponding generating series are obtained in quasi-optimal complexity. This iteration transfers to a numerical scheme that converges unconditionally to the values of the generating series inside their disk of convergence. These results provide important subroutines in random generation. Finally, the approach is extended to combinatorial differential systems.

- Itération de Newton
- Permet évaluation avec convergence quadratique
- Quasi-optimale
- Preuve de correction faisant le pont entre théorie des espèces et classes combinatoires

contrôle de la taille

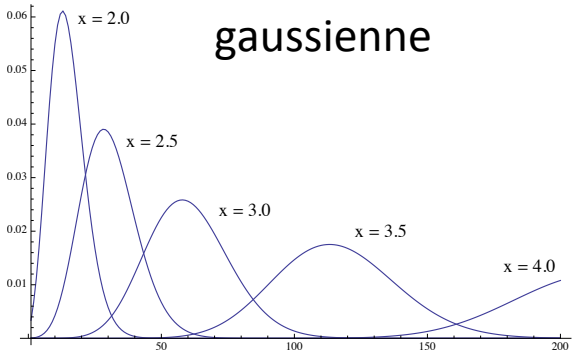
- taille visée par inversion de l'espérance (quand possible)

$$\mathbb{E}_z[N] = z\frac{C'(z)}{C(z)}$$

- différente distribution des tailles, fonction type de singularité

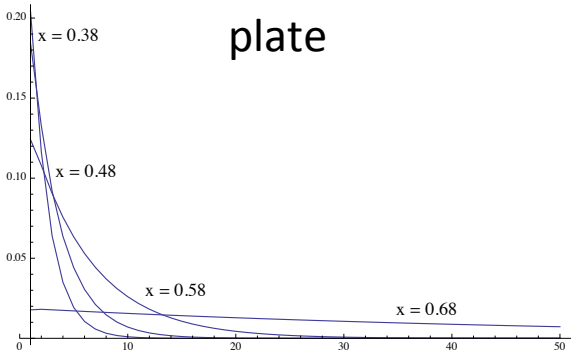
$$C(z) \underset{z\to\rho}{\sim} P(z) + c_0(1-z/\rho)^{-\alpha} + o((1-z/\rho)^{-\alpha}), \quad \alpha \in \mathbb{R}\setminus\{0,-1,-2,\ldots\}$$

- **en taille approchée :** nombre de rejets constant pour tous les cas, sauf le cas "piqué" correspondant à un exposant singulier α négatif



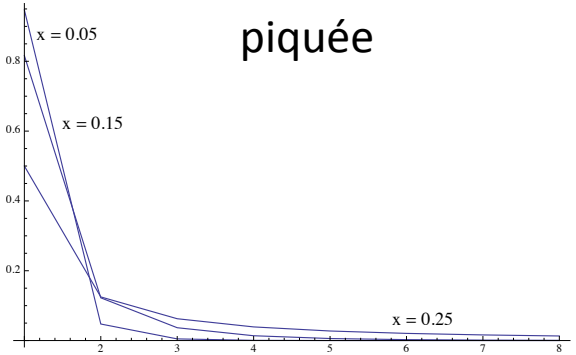
gaussienne

$$P(z) = \mathrm{e}^{\mathrm{e}^z - 1}$$



plate

$$S(z) = \frac{1}{1-(\mathrm{e}^z-2)}$$



piquée

$$B(z) = \frac{1-\sqrt{1-4z}}{2}$$

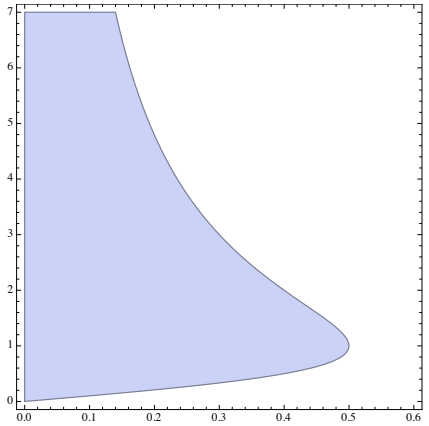
la distribution "piquée"

- Deux méthodes pour la gérer efficacement
  - pointer la classe combinatoire (modifier la distribution des tailles, sans biaiser l'uniformité)
  - **génération singulière** (ne pas viser la taille, mais prendre comme paramètre la singularité $z = \rho$)

- Paradoxalement, la génération singulière est ce qu'il y a de plus facile à faire : recherche de la singularité par dichotomie [Darrasse 2011]

trouver le bon paramètre [Darrasse 2011]

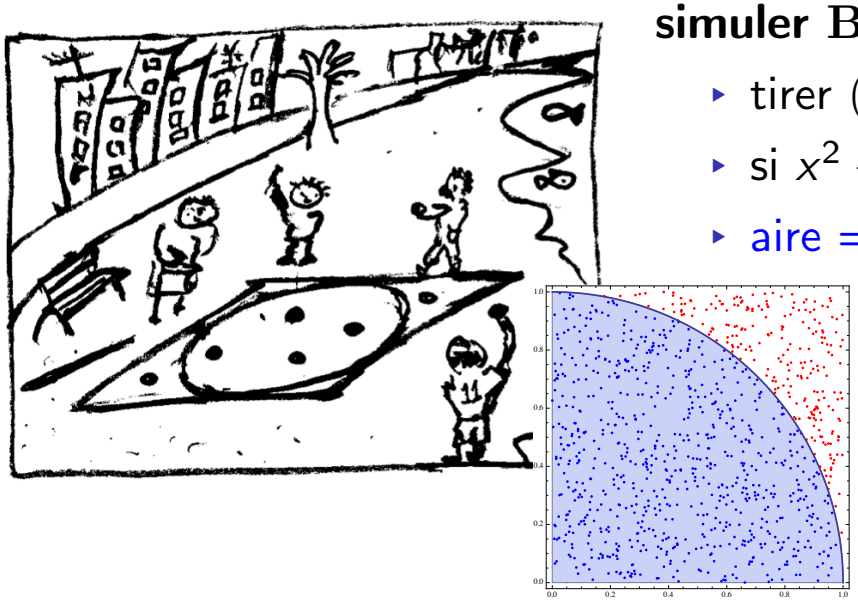
- l'oracle évalue quasi-optimalement
- pour les classes algébriques (avec exposant singulier négatif), génération singulière
  - 2 points de départ $z = 0$ et $z = 1$ (trivialement dans et hors du bidon)
  - si l'oracle diverge, on est en dehors
  - par dichotomie, trouve la singularité



- évaluation faite pour chaque tentative

3. Boltzmann sans évaluation

"remplacer un calcul complexe par quelques calculs simples"



simuler $\mathbf{Ber}(\pi/4)$ (= 1 avec prob. $1/\pi$ et 0 sinon)

- tirer (x, y) uniformément
- si $x^2 + y^2 \leqslant 1$ alors 1 sinon 0
- aire $= \pi/4$ et aire $= 1$ donc prob. $\pi/4$

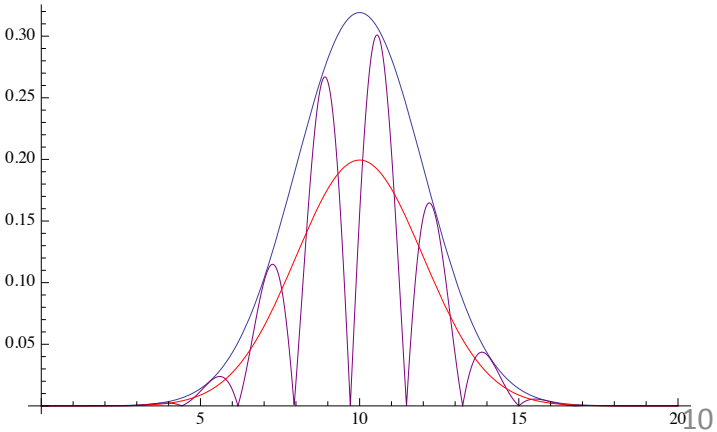
avantages

$\Rightarrow$ pas besoin de connaître π

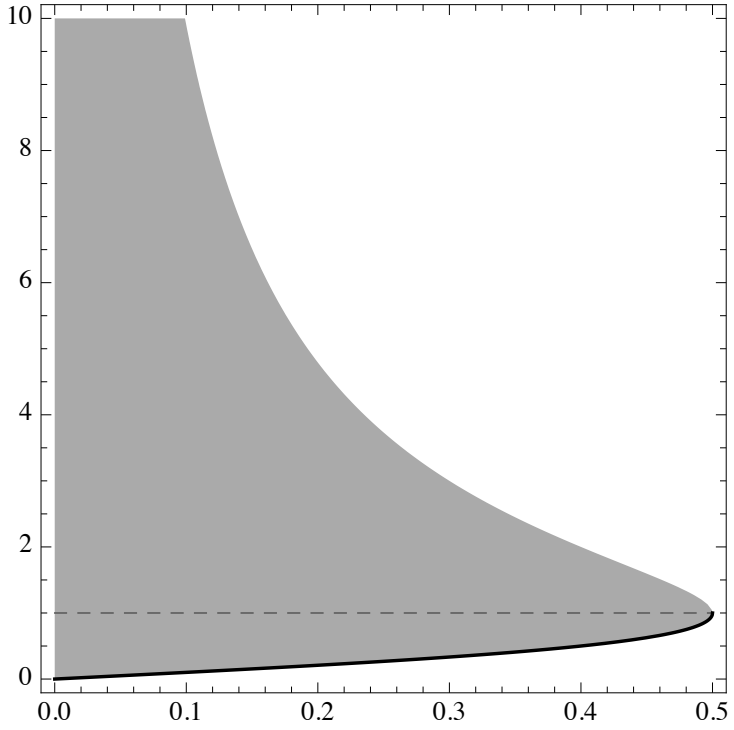
généralisation [Von Neumann 51, Devroye 76]

- distribution complexe $f(x)$
- distribution simple à échantillonner $g(x)$
- mettre à l'échelle, $c \cdot g(x)$, pour recouvrir courbe de $f(x)$
- tirer loi $g(x)$, rejet proport. $f(x)/(c \cdot g(x))$

(complexité: $1/c$ tirages en moyenne)



le “bidon” combinatoire



$B(z) = z + z \cdot B(z)^2 \qquad b \geqslant z + zb^2$

Est-on obligé de rester sur la courbe ?

- Prendre “l’autre” courbe : avoir des objets infinis [Bodini, Moroz, Tafat 2012]
- Prendre toute la zone grise?
  - Est-ce plus “simple” ?
  - Quels objets sortent ?
  - Ou plutôt comment éviter un biaisage des objets générés ?

générateurs analytiques

- Un générateur analytique pour une classe $\mathcal{A}$:

$$\mathbb{P}_{(z,a)}[\alpha \in \mathcal{A}] = \frac{z^{|\alpha|}}{a} \qquad \mathbb{P}_{(z,a)}[\dagger] = 1 - \frac{A(z)}{a}$$

- La normalisation par $a \geqslant A(z)$ qui est une **approximation** de l'évaluation $A(z)$
- Probabilité de **mort** (arrêt immédiat de tout le processus de génération)
- Quand $a = A(z)$: pas d'approx. = pas de mort

proportion d'objets "morts"

Théorème 1: la proportion d'objets rejetés par "mort" ne dépend pas de la taille (celle visée ou de l'objet tiré *in fine*), et est $1 - A(z)/a$

$$\begin{aligned}
\mathbb{E}_{(z,a)}[\#\dagger] &= \sum_{k=0}^{\infty} n \left(1 - \frac{A(z)}{a}\right)^k \frac{A(z)}{a} \\
&= \frac{A(z)}{a} \frac{\left(1 - \frac{A(z)}{a}\right)}{\left(\frac{A(z)}{a}\right)^2} \\
&= \frac{a}{A(z)} \left(1 - \frac{A(z)}{a}\right) \\
&= \frac{a}{A(z)} - 1
\end{aligned}$$

constructeurs (I)

- Notation : $\Gamma\mathcal{A} : [p_1] \cdot \mathrm{Ber}(p_2) \Rightarrow X \mid Y$
  - pour générer la classe $\mathcal{A}$
  - d'abord on échoue avec probabilité $1 - p_1$
  - puis avec probabilité p_2 on renvoie X et sinon Y

constructeurs (II)

- Soit la classe $\mathcal{A} = \mathcal{B} + \mathcal{C}$ avec $a_0 \geqslant b_0 + c_0$, le générateur est

$$\Gamma\mathcal{A} : \left[\frac{b_0 + c_0}{a_0}\right] \cdot \mathrm{Ber}\left(\frac{b_0}{b_0 + c_0}\right) \Rightarrow \Gamma\mathcal{B} \mid \Gamma\mathcal{C}$$

- **Preuve**

$$\mathbb{P}_{(z,a_0)}[\alpha \in \mathcal{A}] = \frac{b_0 + c_0}{a_0}\left(\frac{b_0}{b_0 + c_0}\mathbb{P}_{(z,b_0)}[\alpha \in \mathcal{B}] + \frac{c_0}{b_0 + c_0}\mathbb{P}_{(z,c_0)}[\alpha \in \mathcal{C}]\right)$$

$$\mathbb{P}_{(z,a_0)}[\alpha \in \mathcal{A}] = \frac{b_0 + c_0}{a_0}\left(\frac{b_0}{b_0 + c_0}\frac{z^{|\alpha|}}{b_0} + \frac{c_0}{b_0 + c_0}\frac{z^{|\alpha|}}{c_0}\right) = \frac{z^{|\alpha|}}{a_0}.$$

constructeurs (III)

- Soit la classe $\mathcal{A} = \mathcal{B} \times \mathcal{C}$ avec $a_0 \geqslant b_0 \cdot c_0$, le générateur est

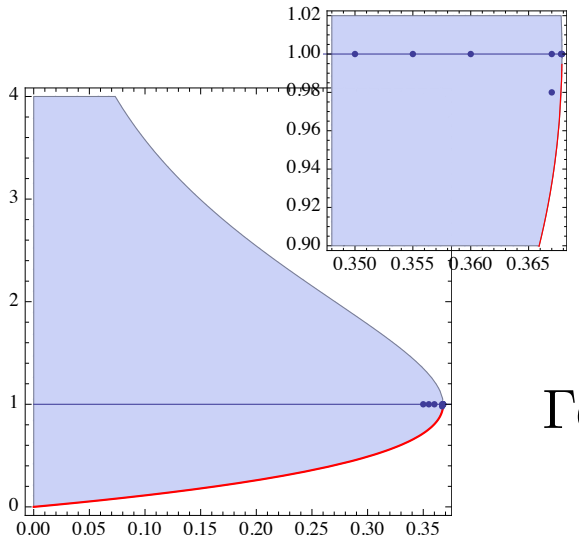
$$\Gamma\mathcal{A} : \left[\frac{b_0 \cdot c_0}{a_0}\right] \Rightarrow (\Gamma\mathcal{B} \ ; \ \Gamma\mathcal{C})$$

- **Preuve :** soit $\alpha = (\beta, \gamma)$

$$\mathbb{P}_{(z,a_0)}[\alpha \in \mathcal{A}] = \frac{b_0 \cdot c_0}{a_0} \mathbb{P}_{(z,b_0)}[\beta \in \mathcal{B}] \, \mathbb{P}_{(z,c_0)}[\gamma \in \mathcal{C}]$$

$$\mathbb{P}_{(z,a_0)}[\alpha \in \mathcal{A}] = \frac{b_0 \cdot c_0}{a_0} \frac{z^{|\beta|}}{b_0} \frac{z^{|\gamma|}}{c_0} = \frac{z^{|\beta|+|\gamma|}}{a_0} = \frac{z^{|\alpha|}}{a_0}.$$

exemple : arbre de Cayley



- L'inéquation :

$$\mathcal{C} = \mathcal{Z} \times \textsc{Set}\,(\mathcal{C}) \quad \Rightarrow \quad C(z) = z \cdot \exp(C(z))$$
$$\Rightarrow \quad c \geqslant z \cdot \exp c$$

- Le générateur :

$$\Gamma\mathcal{C}(z,c) : \left[\frac{z \cdot \exp c}{c}\right] \cdot \mathrm{Poi}(c) \Rightarrow \square(\Gamma\mathcal{C}(z,c), \ldots, \Gamma\mathcal{C}(z,c))$$

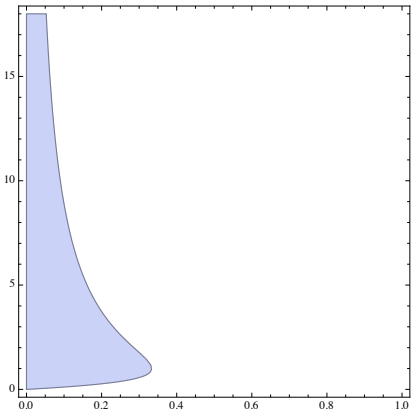
				$c = 1$				$c = 0.98$
z	0.35	0.36	0.367	0.3678	0.36787	0.367879	e^{-1}	0.367
mort observée	28.8%	19.2%	6.4%	1.7%	0.4%	0.3%	0%	3.5%
mort théorique	28.3%	19.4%	6.8%	2.1%	0.7%	0.2%	0%	3.9%
moyenne	6.6	9.9	28.8	127.	177.3	2716.7	4944.3	35.9
max	235	131	1493	17 799	26 531	826 167	2 518 975	1563

4. arbres simples

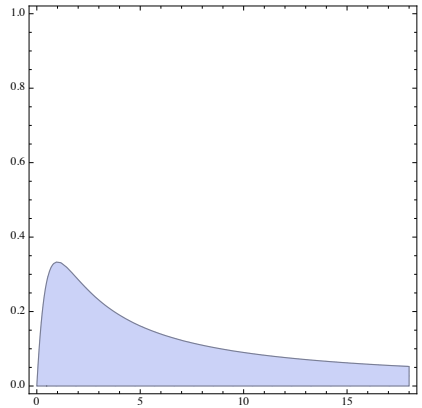
- Arbres avec fonction $\phi(z)$ polynôme de l'arité

$$\mathcal{A} = \mathcal{Z} \times \Phi(\mathcal{A}) \quad \Rightarrow \quad A(z) = z\phi(A(z)) \quad \Rightarrow \quad a \geqslant z\phi(a)$$

- **Idée :** trouver z en fonction de a (pas l'inverse)
- Permet de rapporter le problème à une recherche de maximum local



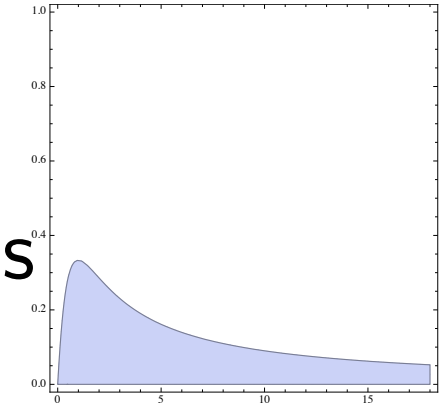
recherche de singularité



recherche de maximum

différence avec méthode traditionnelle

- Les arbres simples sont “piqués” donc génération singulière (on calibre en la singularité et on vise une espérance infinie)
- La recherche de singularité nécessite un nombre logarithmique d’appels à l’oracle
- Recherche de maximum : un nombre logarithmique de modification du paramètre (sans évaluation)



5. arbres d'Otter

- arbres binaires non plans *(= les fils ne sont pas ordonnés)* [Otter 1947]

ANNALS OF MATHEMATICS
Vol. 49, No. 3, July, 1948

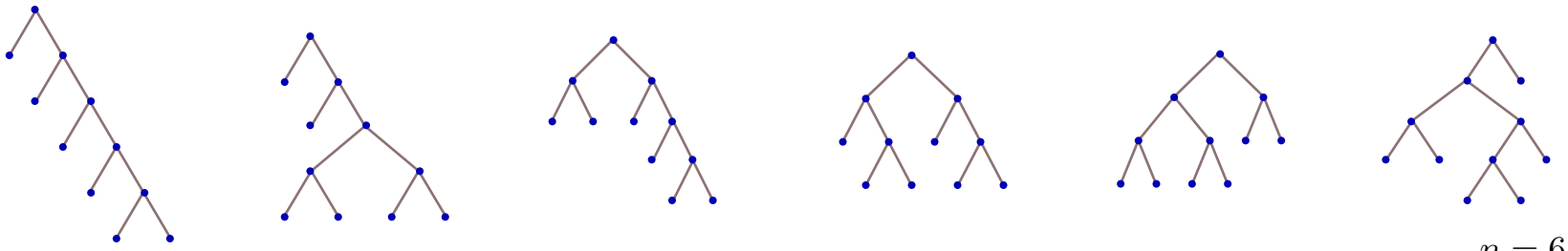
THE NUMBER OF TREES

RICHARD OTTER[1]

(Received June 10, 1947)

$$\mathcal{V} = \mathcal{Z} + \mathrm{MSET}_2(\mathcal{V})$$

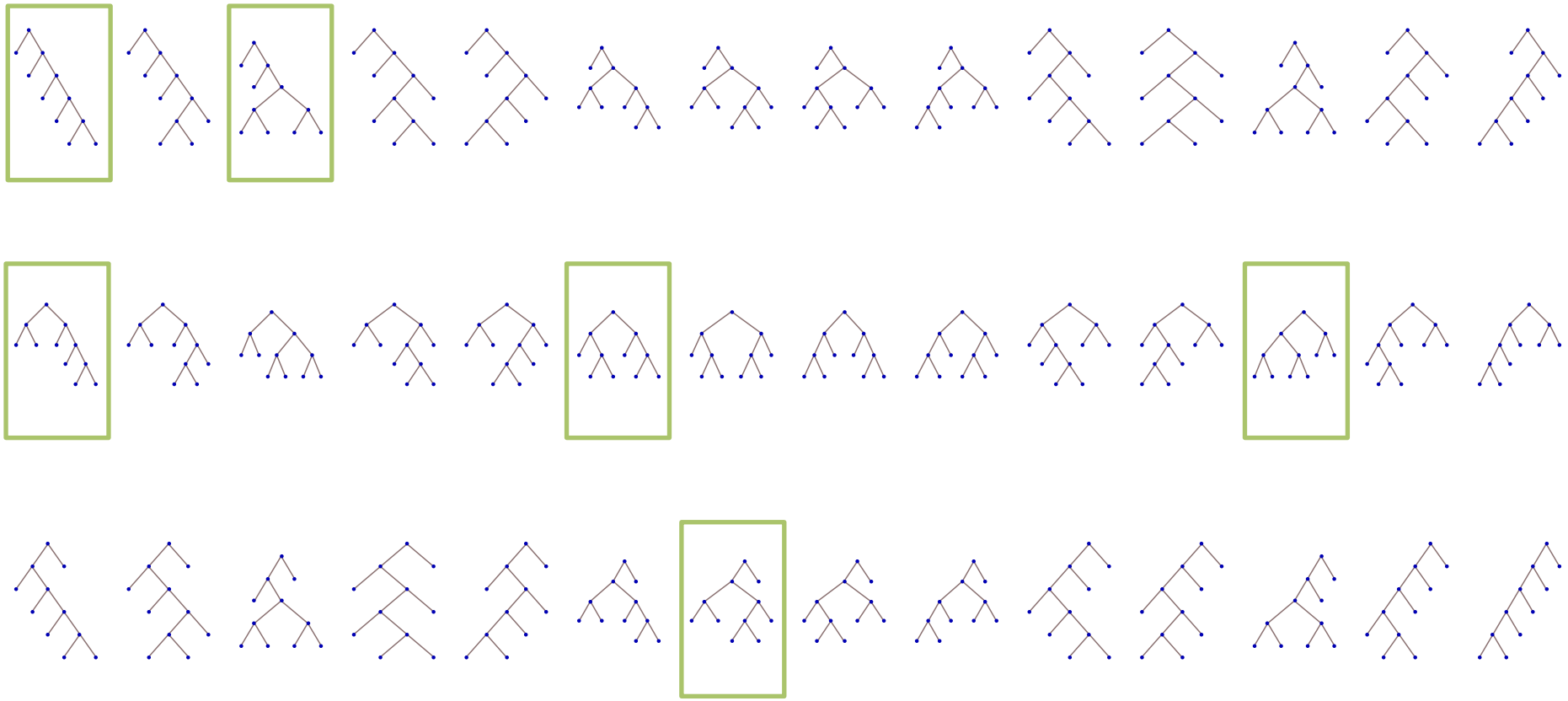
1, 1, 1, 2, 3, **6**, 11, 23, 46, 98, 207, 451, 983,... **A001190**



$n = 6$

plan vs. non-plan

42 arbres binaire plans et **6 non plans**



boltzmann pour les arbres d'otter

arbres symétriques

$$\mathcal{V} = \mathcal{Z} + \mathrm{MSET}_2(\mathcal{V}) \quad \Rightarrow \quad V(z) = z + \frac{1}{2}V(z)^2 + \frac{1}{2}V(z^2)$$

$\Gamma\mathcal{V}(z) :=$ **if** Ber(p_l) $= 1$ **then return** Feuille
else
if Ber($p_b\,/(1-p_l\,)$) $= 1$ **then**
return $\{\Gamma\mathcal{V}(z), \Gamma\mathcal{V}(z)\}$
else
arbre $:= \Gamma\mathcal{V}(z^2)$
return { arbre, arbre }

$$p_l := \frac{z}{V(z)}$$

$$p_b := \frac{1/2V(z)^2}{V(z)}$$

$$p_s := \frac{1/2V(z^2)}{V(z)}$$

- requiert éval. de SG en nb log. de points
- on peut aussi utiliser la propriété [Pivoteau 2009]

$$V(z) = 1 - \sqrt{1 - 2z - V(z^2)}$$

générateur pour les arbres d'otter (I)

- Soit une classe définie par $\mathcal{A} = \mathrm{MSET}_2(B)$
- Donc

$$A(z) = \frac{B(z)^2 + B(z^2)}{2} \quad \Rightarrow \quad a \geqslant \frac{b^2 + b'}{2}$$

- Et le générateur

$$\Gamma\mathcal{A}(z,a) : \left[\frac{b^2+b'}{2a}\right] \cdot \mathrm{Ber}\left(\frac{b^2}{b^2+b'}\right) \Rightarrow \{\Gamma\mathcal{B}(z,b)\ ;\ \Gamma\mathcal{B}(z,b)\} \quad | \quad \Gamma\mathcal{B}(z^2,b') \text{ and duplicate.}$$

probabilité de garder un objet

approximation de valeurs (1)

- L'équation fonctionnelle se transforme

$$V(z) = z + \frac{1}{2}V(z)^2 + \frac{1}{2}V(z^2) \quad \Rightarrow \quad v \geqslant z + \frac{v^2}{2} + \frac{v'}{2}$$

- On pose $v_{[i]} := V(z^{2^i})$
- Par itération, on a les inéquations

$$v_{[i]} \geqslant z^{2^i} + \frac{{v_{[i]}}^2 + v_{[i+1]}}{2}$$

- Terme "constant" est z^{2^i}, on pose $v_{[i]} = K z^{2^i}$

approximation de valeurs (II)

- En insérant dans l'inéquation

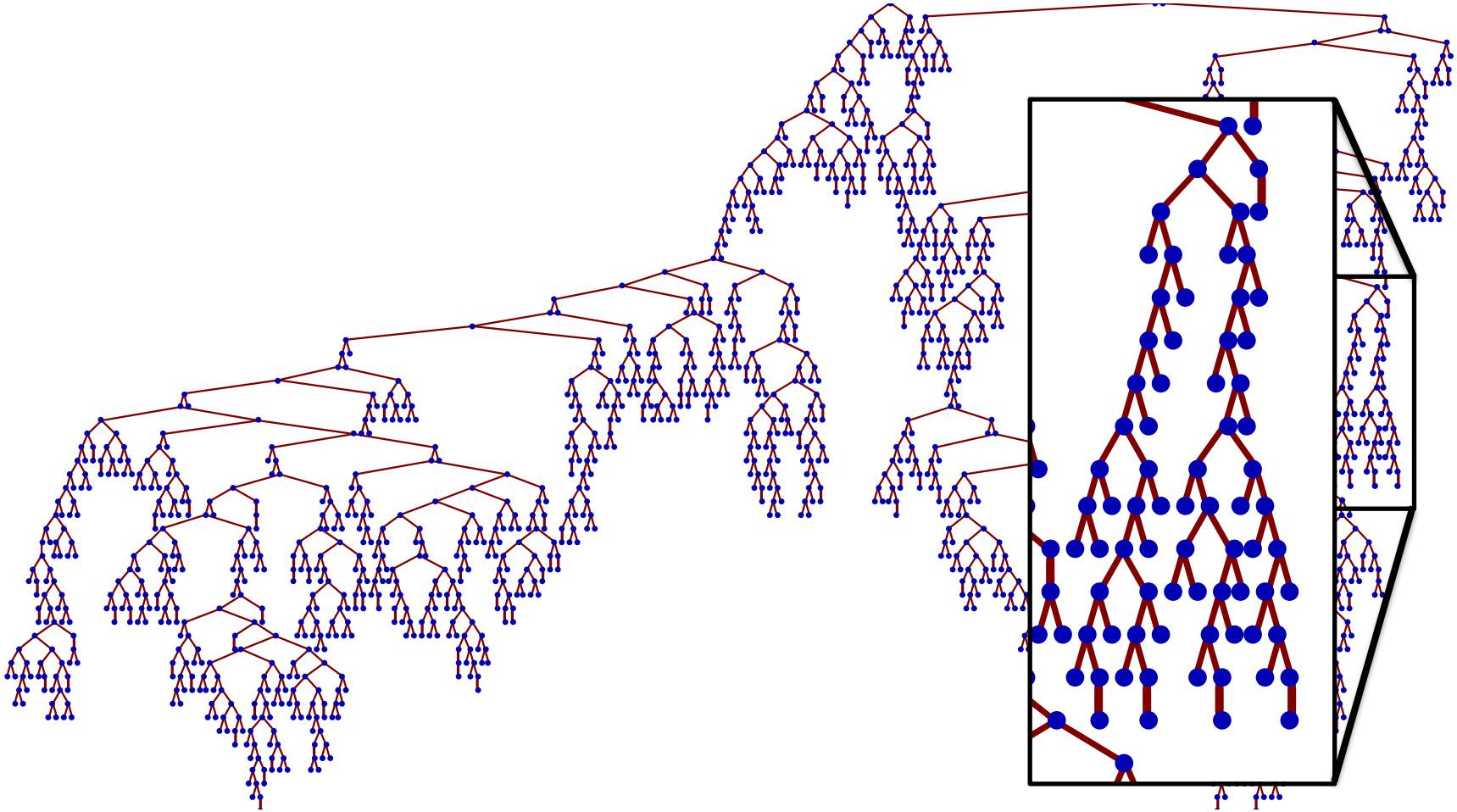
$$K \geqslant 1 + \frac{K z^{2^i}}{2}(K+1)$$

- On fixe un seuil i_0 à partir duquel K est suffisamment petit, e.g. 1,1 ($K > 1$) puis on résout pour $i < i_0$
- Pour $i \geqslant i_0$, on approxime avec l'inéquation

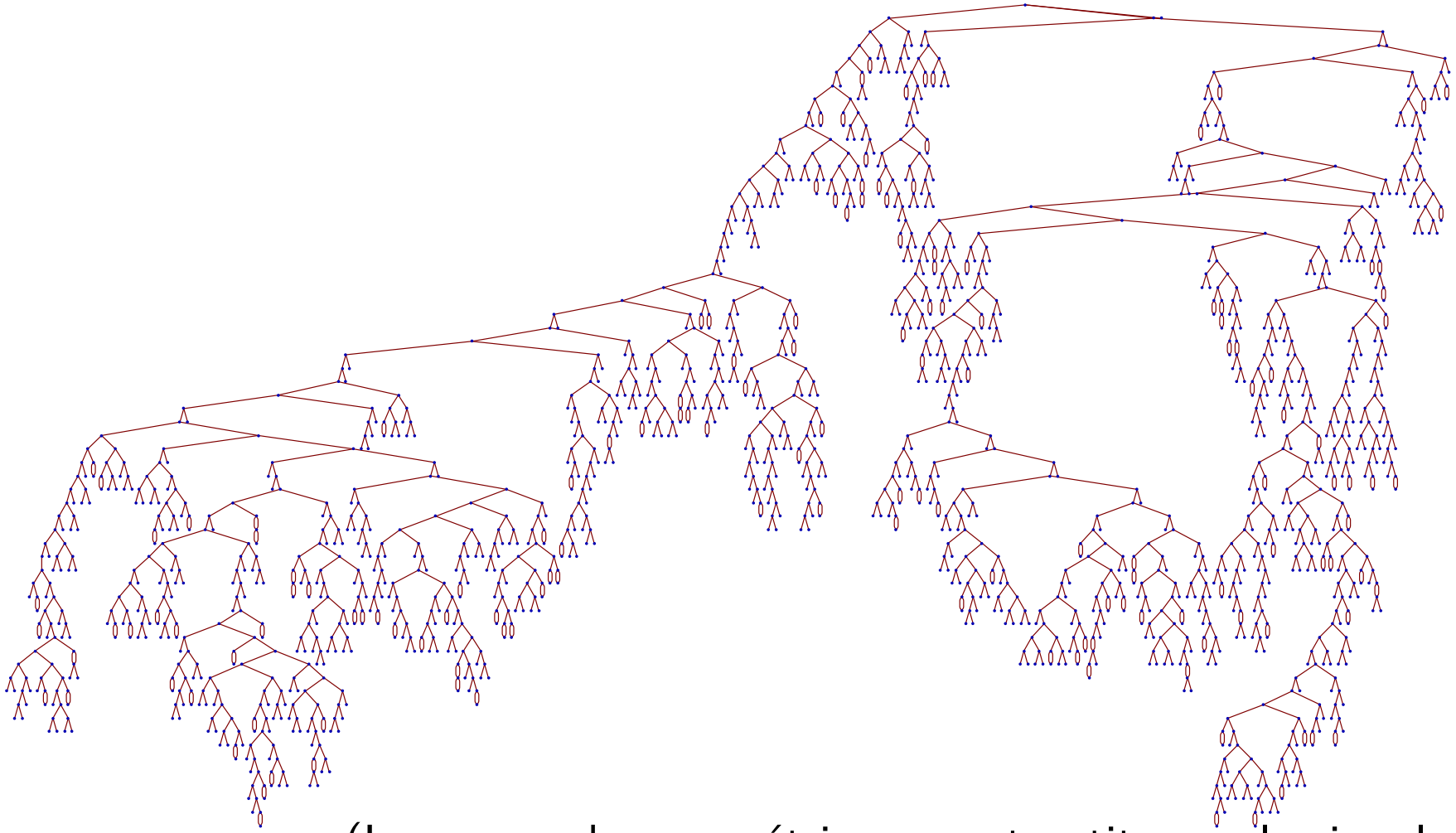
```
>>> v_cache = calculer_vi(compz, 10)
[0.99999999990347351441, 0.19460499265711741805, 0.027008323220737167869,
 0.00069204119151622108140, 4.7825906049817228644e-7, 2.2873151016281242364e-13,
 5.2318103741336829440e-26, 2.7371839790892827503e-51, 7.4921761353830390684e-102,
 5.6132703243603130559e-203, 3.1508803734344134721e-405]

>>> def v(i, z) :
...   if i < 10 :
...     return v_cache[i]
...   else:
...     return (z^(2^i) + 1.1*z^(2^(i+1)))
```

un arbre d'otter de taille 1889



un arbre d'otter de taille 1889

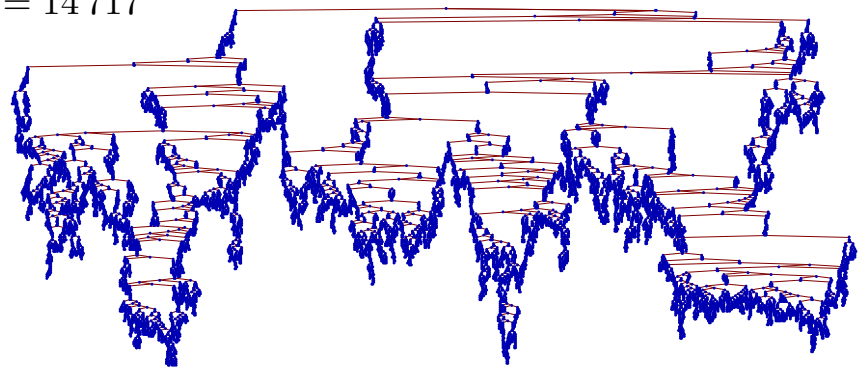


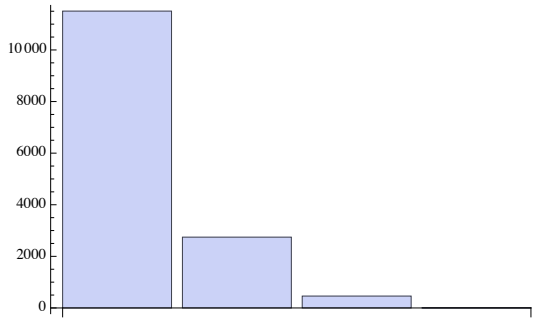
(Les sous-arbres symétriques sont petits, car la singularité dominante est celle du terme non-symétrique.)

grands arbres d'otter

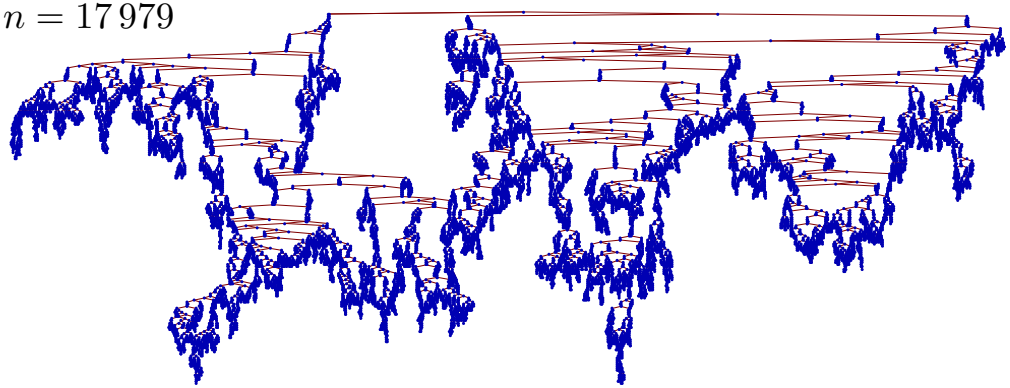
grands arbres générés aléatoirement +
distribution des nœuds par degré de symétrie

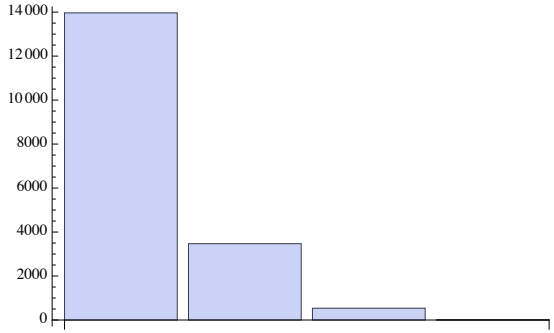
$n = 14\,717$





$n = 17\,979$





6. conclusion

- Génération de Boltzmann avec valeurs arbitraires au lieu de la série génératrice

- **Avantages**
  - Principe global permettant de garantir une génération exacte en présence d'approximations
  - Dans le cas des arbres simples :
    - recherche de singularité à celle de maximum
    - évite les sur-appels à l'oracle requits par la génération singulière
  - Dans certains cas (arbres d'Otter) : s'affranchir d'évaluation quand il en faut plus qu'un nombre constant

- **Limites**
  - Pas encore de simulations pour déterminer le gain en efficacité
  - Les classes, pour lesquelles "l'automatisation" de la détermination des paramètres est réglée, ne sont pas encore identifiées