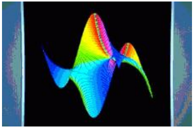


Boltzmann Sampling and Random Generation of Combinatorial Structures

Philippe Flajolet

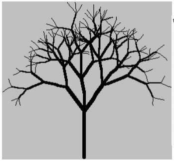
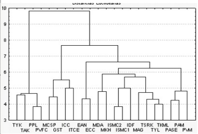
Based on joint work with *Philippe Duchon, Éric Fusy, Guy Louchard, Carine Pivoteau, Gilles Schaeffer*

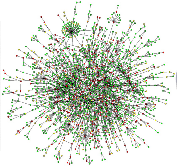
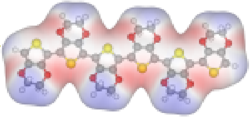
GASCOM'06, Dijon, September 12, 2006

$\mathcal{C}$ is a class of combinatorial structures.
$\mathcal{C}_n$ = collection of objects of size n.

Draw uniformly at random from $\mathcal{C}_n$?: $\boxed{\mathbb{P}(\gamma) = \frac{1}{C_n}, \quad C_n := \|\mathcal{C}_n\|.}$

E.g.: trees, permutations, words, graphs, mappings, maps, etc.

Classification theory [Van Cutsem]; image synthesis [Viennot]; random testing in software eng. [J. Fayolle], combinatorics; simulation & statistical analysis of models in genetics [Denise], ecology [de Reffie], ...

Random Generation and Combinatorics

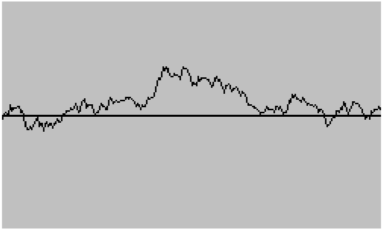
- *Bijective method*: find bijection with simpler (product) set.
- *Surjective method*: find a “multiple” set that is simpler
- *Rejection method*: find a larger set and filter.
- *Markov method*: superimpose Markov chain structure & travel!
- *Recursive method*: decompose according to counting probabilities
- **Boltzmann**: This talk!

Bijective method

Find bijection with simpler set

Class $\mathcal{C}$ is such that $C_n = \|\mathcal{C}_n\|$ is a product.

Words: $\mathcal{W}_n \cong \{a, b\}^n \Longrightarrow n$ random flips.
Permutations: $\mathcal{P}_n \cong [0] \times [0\,..\,1] \times \cdots \times [0\,..\,n-1] \Longrightarrow n$ RVs

Dyck bridges: $\mathcal{B}_{2n} \cong \binom{2n}{n}$:  [Vitter]

Usually requires pure product form!

Surjective method

Find many-to-one uniform *correspondence between $\mathcal{C}_n$ and* simpler *set $\mathcal{A}_n$.*

divisibility: $C_n \mid A_n$.

Dyck excursions: by *conjugacy* with bridges $\rightsquigarrow$ **Catalan trees**.

$$C_n = \frac{1}{2n+1}\binom{2n+1}{n}.$$

Jean-Luc Rémy's algorithm for **binary trees**.

Planar maps: cf Schaeffer *et al.*: by tree conjugation.

⚠ Usually requires pure product form!

Rejection method

Find larger set *such that* $\mathcal{C}_n \subset \mathcal{D}_n$, *with* simpler $\mathcal{D}$

$\Longrightarrow$ `Draw` $\delta \in \mathcal{D}$. `Test whether` $\delta \in \mathcal{C}$; repeat if needed

Problem: Probability of success is $\dfrac{C_n}{D_n}$.

E.g. **Prime numbers; irreducible polynomials**. Cf Ruskey.

E.g. Florentine algorithm for **Dyck/Motzkin meanders**.

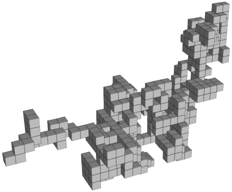
⚠ Avoid exponentially small probabilities?

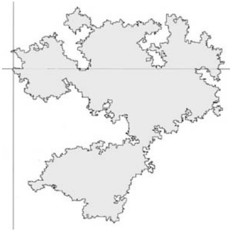
Markov method

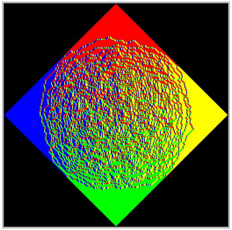
— *View elements of a class $\mathcal{S}_n$ as states of a* **Markov chain**

— Set up transitions (e.g, via transformations)

If the graph is regular, then the stationary distribution is uniform.







Reversible Markov chains, Coupling [Propp-Wilson, Jerrum,. . .].

$\rightsquigarrow$ **Self-avoiding walks, dimer coverings**, "hard" combinatorial objects.

May need information on mixing speed λ_2.

Recursive method

- *Use counting sequences to decide splitting probabilities.*

E.g.: **Binary trees** with n external nodes, class $\mathcal{B}_n$.

— A. Set up recurrence $B_n = \sum_{k=1}^{n-1} B_k B_{n-k}$.

— B. Split $n \mapsto \langle k, n-1-k \rangle$ with probability $\frac{B_k B_{n-k}}{B_n}$.

Theorem (Recursive method)

Complexity of preprocessing is $O(n^2)$ large integer operations. Complexity of boustrophedonic random generation is $O(n \log n)$ arithmetic operations.

- **ECO systems.** • Wilf's **path approach.**

J. van der Hoeven: Preprocessing in time $O(n^{1+\varepsilon})$. A. Denise & P. Zimmermann: Floating point implementations. Also: MAPLE `Combstruct`.

Boltzmann framework

Principle:

• Generate according to a **distribution spread over all $\mathcal{C}$**, depending on **control parameter x.**
• Size becomes a random variable (RV).
• Target choice of x to get objects of size near n with fair probability.

Cf Statistical Physics: $\mathbb{P}(\gamma) = \frac{1}{Z} \exp\left(-\frac{\beta}{T} E[\gamma]\right)$.

Ordinary (unlabelled) Boltzmann models

Assign to $\gamma \in \mathcal{C}$ probability proportional to exponential of its size:

$$\mathbb{P}(\gamma) \propto x^{|\gamma|} \qquad \Longrightarrow \qquad \boxed{\mathbb{P}(\gamma) = \frac{x^{|\gamma|}}{C(x)}},$$

$C(x) = \sum_n C_n x^n$ is ordinary generating function (OGF).
Requires $x \leq \rho_C$, where ρ_C is the radius of convergence of $C(x)$.

$\rightsquigarrow$ Size becomes a random variable:

$$\mathbb{P}(\text{Size} = n) = \frac{C_n x^n}{C(x)}.$$

Boltzmann Samplers: the Plan!

Develop design rules given combinatorial specifications.

— Basic constructions: $\cup, \times, \mathrm{SEQ}$
— Labelled models: add $\mathrm{SET}, \mathrm{CYC}$
— Return to unlabelled models: add $\mathrm{MSET}, \mathrm{PSET}, \mathrm{CYC}$

Do optimization w.r.t. size at the end: complexity issues.

Based on [DuFlLoSc04] in CPC for labelled; [FlFuPi06] for unlabelled.
Cf. F.+Sedgewick, *Analytic Combinatorics*.

Unions, products

Lemma (Disjoint unions)

Boltzmann sampler ΓC *for* $\mathcal{C} = \mathcal{A} \cup \mathcal{B}$*:*
`With probability` $\frac{A(x)}{C(x)}$ `do` $\Gamma A(x)$ `else do` $\Gamma B(x)$

Lemma (Products)

Boltzmann sampler ΓC *for* $\mathcal{C} = \mathcal{A} \times \mathcal{B}$*:*
`Generate` *`independent`* `pair` $\langle \Gamma A(x), \Gamma B(x) \rangle$.

Proofs $=$ *One-liners!* Using basic definitions of probability.

— Disjoint union: $|\gamma| = n \Longrightarrow$ if $\gamma \in \mathcal{A}$ then $\mathbb{P}_{\mathcal{C}}(\gamma) = \frac{x^n}{A(x)} \cdot \frac{A(x)}{C(x)} \ldots$

— Product: $\mathbb{P}_{\mathcal{C}}(\gamma) = \frac{x^k}{A(x)} \cdot \frac{x^{n-k}}{B(x)} = \frac{x^n}{C(x)}$.

Sequences

Lemma (Sequences)

Boltzmann sampler ΓC *for* $\mathcal{C} = \textsc{Seq}(\mathcal{A})$*:*

- `Generate` K `which is` *`geometric`* `with parameter` $A(x)$
- `Generate` *`independent`* K`-tuple` $\langle \Gamma A(x), \ldots, \Gamma A(x) \rangle$`.`

Proof. Recursive equation: $\mathcal{C} = \mathbf{1} + \mathcal{A}\mathcal{C}$ with $+, \times$ constructions.

With probability $\dfrac{1}{A(x)}$ STOP; else $\Gamma A(x)$ and continue rec. with $\Gamma C(x)$.

Number of trials of Bernoulli RV till success is Geometric.

Specifications with $\{\cup, \times, \textsc{Seq}\}$

Specs	*GF*	*Sampler*
$\mathbf{1}$ or $\mathcal{Z}$ (atom)	1 or x	$\Gamma C :=$ `output` $\mathbf{1}$ `or` $\bullet$
$\mathcal{C} = \mathcal{A} \cup \mathcal{B}$	$C(x) = A(x) + B(x)$	$\Gamma C(x) := \frac{A(x)}{C(x)} \longrightarrow \Gamma B(x) \mid \Gamma C(x)$
$\mathcal{C} = \mathcal{A} \times \mathcal{B}$	$C(x) = A(x) \times B(x)$	$\Gamma C(x) := \langle \Gamma B(x), \Gamma C(x) \rangle$
$\mathcal{C} = \textsc{Seq}(\mathcal{A})$	$C(x) = \dfrac{1}{1 - A(x)}$	$\Gamma C(x) := \mathrm{Geom}[A(x)] \Longrightarrow \Gamma A(x)$

Compile sampler from specification automatically.

Specifications with $\{\cup, \times, \textsc{Seq}\}$ — continued

Theorem (Complexity Minitheorem)

Given **oracle** *that provide the finitely many values of GFs, complexity is* **linear** *in size of object produced.*

Proof $\{\cup, \times, \textsc{Seq}\}$: overhead $O(1)$ per node of derivation tree. Complexity model: exact computations over $\mathbb{R}$; in practice, "floats" (more later).

Definition

Regular specification = iterative (nonrecursive) with $\{\cup, \times, \textsc{Seq}\}$.
Contex-free specification = recursive with $\{\cup, \times, \textsc{Seq}\}$.

Proposition

Regular structures *and* **context-free structures** *have Boltzmann samplers of linear-time complexity.*

Specifications with $\{\cup, \times, \textsc{Seq}\}$ — continued (2)

Regular specifications

- **Binary words** with **longest run** of a's of length < 17.

$$\textsc{Seq}_{<17}(\{a\}) \cdot \textsc{Seq}\left(b\,\textsc{Seq}_{<17}(\{a\})\right).$$

- **Codes**, e.g., $\{aba, abaaa, abba\}$.
- **Polyominos** that have rational GF, e.g., Vertically convex.

- **Languages** recognized by deterministic **finite automata** E.g., Strings containing three times the pattern "abracadabra".
- **Paths in digraphs** even in the presence of **sinks**.

Specifications with $\{\cup, \times, \textsc{Seq}\}$ — continued (3)

Contex-free specifications.

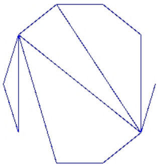
- **Binary trees**: $\mathcal{B} = \mathcal{Z} + \mathcal{B} \times \mathcal{B}$.

— Solve quadratic equation $B = x + B^2$ numerically, given x;
— Out put single node with probability $\frac{x}{B}$;
Else: Do two independent recursive calls to $\Gamma B(x)$.

For **rooted unlabelled trees**, Boltzammn model reduces to branching process.

Generate **Motzkin trees** [$\neq$Alonso-Schoot], (unbalanced) **2–3-trees; random walks with finite step sets (dice)**, etc.

Noncrossing graphs:



Exponential (labelled) Boltzmann models

- For labelled classes, model is called

exponential or labelled Boltzmann model

$$\mathbb{P}(\gamma) \propto \frac{x^{|\gamma|}}{|\gamma|!} \qquad \Longrightarrow \qquad \boxed{\mathbb{P}(\gamma) = \frac{1}{C(x)}\frac{x^{|\gamma|}}{|\gamma|!}},$$

$C(x) := \sum_n C_n \frac{x^n}{n!}$ is exponential GF (EGF).

— Replace Cartesian product by **labelled product** (distribute labels).
— **Unions, products, sequences**: work like before, *but* with EGFs.
— **Sets and cycles** = to do!

Labelled sets and cycles

Poisson law: $\mathbb{P}(X = k) = e^{-\lambda}\frac{\lambda^k}{k}!$.

Logarithmic law: $\mathbb{P}(X = k) = \frac{1}{L}\frac{\lambda^k}{k}$, $L := 1/\log(1-\lambda)^{-1}$.

Lemma

Labelled sets and labelled cycles are obtained by a Poisson and Logarithmic generator resp.

$$\begin{array}{lcl} \mathcal{C} = \textsc{Set}(\mathcal{A}) & : & \mathrm{Pois}(A(x)) \Longrightarrow \Gamma A(x) \\ \mathcal{C} = \textsc{Cyc}(\mathcal{A}) & : & \mathrm{Loga}(A(x)) \Longrightarrow \Gamma A(x) \end{array}$$

Cf: $\mathcal{C} = \textsc{Seq}(\mathcal{A}) \quad : \quad \mathrm{Geom}(A(x)) \Longrightarrow \Gamma A(x)$.

Applies to any specifiable class of combinatorial objects

- For each x, need *finite # of computable real constants*.
- Linear-time random generation.
- Size is not controlled (yet)

Example: **Cayley trees** $= \mathcal{T} = \mathcal{Z} \star \mathrm{SET}(\mathcal{T})$.

— Solve $T(x) = xe^{T(x)}$ numerically.

— Generate root $(\mathcal{Z})$;

— Choose random root degree as $\Delta := \mathrm{Pois}(T(x))$;

— Call Δ independent copies of $\Gamma(x)$;

— Hope for the best regarding size ($\rightsquigarrow$ later)

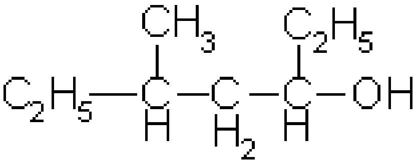
Examples:

Set partitions. $\mathcal{S} = \textsc{Set}(\textsc{Set}_{\geq 1}(\mathcal{Z}))$.
$\#$ components is $\mathsf{Pois}(e^x - 1)$; each comp. is $\mathsf{Pois}(x) \mid \geq 1$ = Vershik.

Ordered set partitions. Geometric triggers Poisson.

Assemblies of filaments. Poisson triggers geometric.

Unlabelled sets and cycles



[Pólya] Carbon has valency 4; hydrogen has valency 1. How to generate a **random alcohol**?.

= **Nonplane unlabelled tree with node degrees** $\in \{0, 3\}$**.**

Need to take care of symmetries to generate object only once!

Unlabelled sets

- The **multiset** construction $\mathcal{C} = \mathrm{MSET}(\mathcal{A})$: form all finite multisets,

$$\mathcal{C} \cong \prod_{\alpha \in \mathcal{A}} \mathrm{SEQ}(\{\alpha\}).$$

(i) *Gedanken Alg.* Scan $\mathcal{A}$ & generate α with multiplicity $\mathrm{Geom}(x^{|\alpha|})$.
(ii) Observe GF equation: $C(x) = \exp(A(x)) \cdot \exp(\frac{1}{2}A(x^2)) \cdots$.
(iii) Do Poisson-controled generator for $\mathcal{A}$ with parameter $A(x)$; repeat with $\frac{1}{2}A(x^2)$; etc.
(iv) Compute when to stop. Collect multset.

Proof involves $\mathrm{Geom}(\lambda) \equiv \mathit{Pois}(\lambda) + \mathit{Pois}(\frac{1}{2}\lambda^2) + \cdots$.

Powersets and cycles

- The **cycle** construction: proceed from GFs. For $\mathcal{C} = \textsc{Cyc}(\mathcal{A})$,

$$C(z) = \log \frac{1}{1 - A(z))} + \frac{1}{2} \log \frac{1}{1 - A(z^2))} + \cdots$$

Treat as infinite union, cf multisets. E.g., **Necklaces**.

- The **powerset** construction $\mathcal{C} = \textsc{PSet}(\mathcal{A})$: form all finite sets (no repetition!). Use identity $1 + z = \frac{(1-z^2)}{(1-z)}$.
Generate Boltzmann multiset and throw away all elements of even multiplicity.

- **Relativized constructions** like $\mathcal{C} = \textsc{MSet}_3(\mathcal{A})$: do $\Gamma A(x^3)$, etc.

Unlabelled constructions

Theorem (Main Complexity Theorem)

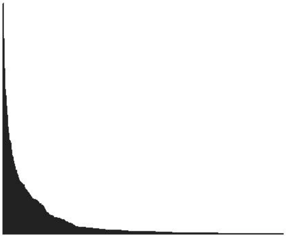
For a class $\mathcal{C}$ specified (poss. recursively) from finite sets using

$$+,\ \times,\ \textsc{Seq},\ \text{MS}\textsc{et},\ \text{MS}\textsc{et}_k,\ \textsc{Cyc},\ \textsc{Cyc}_k,$$

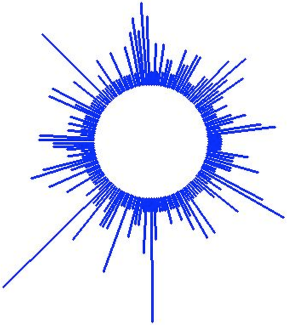
The Boltzman sampler $\Gamma C(x)$ operates in linear time in the size of the object produced.

Also allow for powersets as soon as $\rho < 1$.

Examples. **Integer partitions, nonplane unlabelled trees, alcohols, mapping patterns [functional graphs], series-parallel circuits, etc**

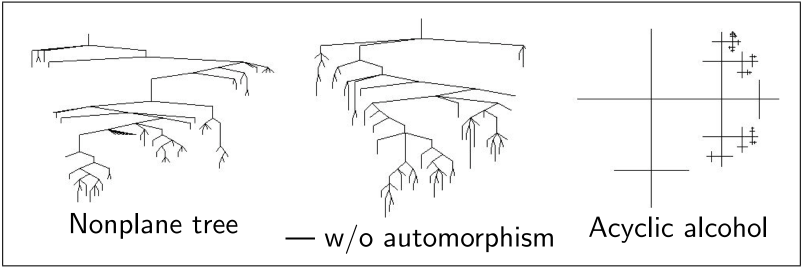


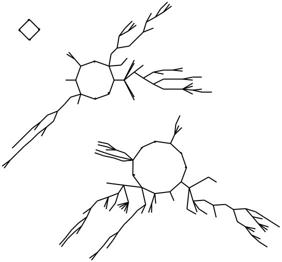
Partition of integer



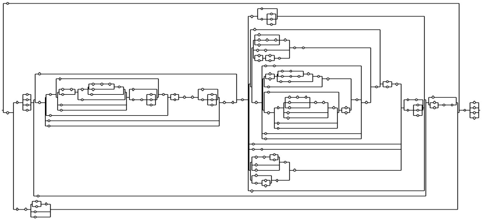
Cyclic composition

Partition of integer into distinct summands


Nonplane tree
— w/o automorphism
Acyclic alcohol



Functional graph



Series-parallel circuit

Complexity

- Size control

$$\mathrm{PGF}(\mathrm{Size}) = \frac{C(ux)}{C(x)} \qquad \Longrightarrow \qquad \mathbb{E}_x(\mathrm{Size}) = \frac{xC'(x)}{C(x)}.$$

Usually requires $x \to \rho_C$ to get large structures.

Size control (1)

- **Free Boltzmann samplers**: produce objects with randomly varying sizes!
  E.g., VC-polyominos: 37, 158, 389, 91, 21, 110, . . .

Size control (1)

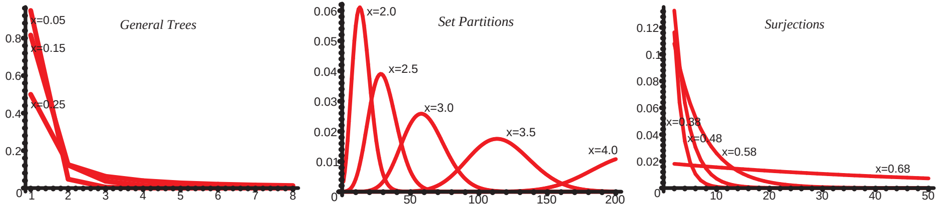
- **Free Boltzmann samplers**: produce objects with randomly varying sizes!
  E.g., VC-polyominos: 37, 158, 389, 91, 21, 110, ...
- **Tuned Boltzmann samplers**: choose x so that expected size $= n$.

Size control (1)

- **Free Boltzmann samplers**: produce objects with randomly varying sizes!
  E.g., VC-polyominos: 37, 158, 389, 91, 21, 110, . . .
- **Tuned Boltzmann samplers**: choose x so that expected size $= n$.
- **Analysis of size distribution** of free sampler determines **complexity**.

Size control (2)

“Frequent” profiles: [cf *Analytic Combinatorics*]



Peaked **Bumpy** **Flat**

Depends on *singularity type* of generating function.

Theorem (Complexity I)

"Bumpy type" is granted for Hayman-admissible models.
Approximate-size complexity $= \mathcal{O}(n)$. *Exact size* $= o(n^2)$.

Applies to GFs that are of type Exp $\circ$ Fast-growth.

Theorem (Complexity II)

"Flat type" is granted for algebraic-logarithmic sing. + infinite
Approximate-size complexity $= \mathcal{O}(n)$. *Exact-size* $= o(n^2)$.

Theorem (Complexity III)

For "critical sequences":
Exact-size complexity $= \mathcal{O}(n)$.

Renewal type of algorithm at critical ρ.

Size control (3): Pointing

Pointing: If $\mathcal{A}$ is a class, then $\mathcal{C} = \mathcal{A}^\bullet$ is the set of objects with one atom pointed, and

$$C_n = nA_n, \qquad C(z) = z\frac{d}{dz}A(z).$$

Uniformity at given size is preserved (only size profile is altered).
Transforms peaked (inefficient) distributions to flat (efficient).
E.g., **binary trees** $\mathcal{B}$:

$$\mathcal{B} = \mathcal{Z} + \mathcal{B} \times \mathcal{B} \quad \Longrightarrow \quad \begin{cases} \mathcal{B} & = \quad \mathcal{Z} + \mathcal{B} \times \mathcal{B} \\ \mathcal{B}^\bullet & = \quad \mathcal{Z} + \mathcal{B}^\bullet \times \mathcal{B} + \mathcal{B} \times \mathcal{B}^\bullet. \end{cases}$$

All simple families of trees: it works!

Discrete samplers

- Real arithmetics versus bit [boolean] complexity?

— Do bit-level generators for Bernoulli, Geometric, Poisson, Logarithmic.

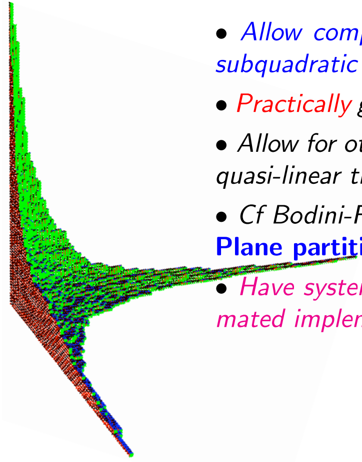
$$\frac{1}{\pi} = \langle 0.010100010111110011000 \rangle_2.$$

Bernoulli: return bit at position Geom($\frac{1}{2}$); Geometric: iterate till 1.
Cf. Knuth-Yao (1976); Von Neumann. Soria-Pelletier et al.

— Integrated samplers for set partitions, etc? Expect *low bit-complexity*!

- In practice do 40D evaluations of constants and be happy!

Conclusions



- *Allow computation over the reals and get linear or subquadratic time samplers.*
- *Practically get objects of sizes in the range* 10^4 *to* 10^8.
- *Allow for other operations: Fusy =* **planar graphs** *in quasi-linear time* $\ll$ *[Noy-Gimenez]*
- *Cf Bodini-Fusy-Pivoteau; Bassino-Nicaud [Nancy]*
Plane partitions; random automata, ...
- *Have systematic design principles! Get largely automated implementations?*

A plane partion of size 15,000 [Carine Pivoteau]

Some literature (all on the web!)

[DuFlLoSc04] "Boltzmann Samplers for the Random Generation of Combinatorial Structures", by Philippe Duchon, Philippe Flajolet, Guy Louchard, Gilles Schaeffer. In *Combinatorics, Probability, and Computing*, Special issue on Analysis of Algorithms, 2004, Vol. 13, No 4–5, pp. 577-625.
[FlFuPi06] "Boltzmann Sampling of Unlabelled Structures", by Philippe Flajolet, Éric Fusy, and Carine Pivoteau. 14 pages. Submitted to *ANALCO'07*.
[BaNi06] "Accessible and deterministic automata: enumeration and Boltzmann samplers". F. Bassino C. Nicaud. In *Fourth Colloquium on Mathematics and Computer Science*.
[Fusy05] "Quadratic exact-size and linear approximate-size random sampling of planar graphs", by Éric Fusy. In *2005 International Conference on Analysis of Algorithms*. *DMTCS* Conference Volume AD (2005), pp. 125-138.
[BoFuPi06] "Random sampling of plane partitions". By Olivier Bodini, Éric Fusy, and Carine Pivoteau. GIn *ASCOM-2006*.