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 $\mathcal{C}_n$?:

$$\mathbb{P}(\gamma) = \frac{1}{C_n}, \quad C_n := ||\mathcal{C}_n||.$$

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 $|C_n|$ is a product.

**Words:** $\mathcal{W}_n \cong \{a, b\}^n \Rightarrow n$ random flips.

**Permutations:** $\mathcal{P}_n \cong [0] \times [0..1] \times \cdots \times [0..n-1] \Rightarrow n$ RVs

**Dyck bridges:** $\mathcal{B}_{2n} \cong \binom{2n}{n}$:

![Graph](image.png)

[Note: Vitter]

⚠️ Usually requires pure product form!
Surjective method

Find many-to-one uniform correspondence between \( C_n \) and simpler set \( A_n \).

\[
divisibility: \quad C_n \mid A_n.
\]

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 $C_n \subset D_n$, with simpler $D$.

$\implies$ Draw $\delta \in D$. Test whether $\delta \in C$; repeat if needed.

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

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 $S_n$ 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, …].

⇒ Self-avoiding walks, dimer coverings, “hard” combinatorial objects.

⚠ May need information on mixing speed $\lambda_2$. 
Recursive method

- **Use counting sequences to decide splitting probabilities.**

  E.g.: Binary trees with $n$ external nodes, class $B_n$.

  - 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** $C$, 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: $\mathbb{P}(\gamma) = \frac{1}{Z} \exp \left( -\frac{\beta}{T} E[\gamma] \right)$. 
Assign to $\gamma \in C$ probability proportional to exponential of its size:

$$\mathbb{P}(\gamma) \propto x^{\lvert \gamma \rvert} \implies \mathbb{P}(\gamma) = \frac{x^{\lvert \gamma \rvert}}{C(x)},$$

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

Size becomes a random variable:

$$\mathbb{P}({\text{Size}} = n) = \frac{C_n x^n}{C(x)}.$$
Develop design rules given combinatorial specifications.

— Basic constructions: $\cup, \times, \text{SEQ}$
— Labelled models: add $\text{SET}, \text{Cyc}$
— Return to unlabelled models: add $\text{MSet}, \text{Pset}, \text{Cyc}$

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)

\[ \text{Boltzmann sampler } \Gamma C \text{ for } C = A \cup B: \]
With probability \( \frac{A(x)}{C(x)} \) do \( \Gamma A(x) \) else do \( \Gamma B(x) \)

Lemma (Products)

\[ \text{Boltzmann sampler } \Gamma C \text{ for } C = A \times B: \]
Generate independent pair \( \langle \Gamma A(x), \Gamma B(x) \rangle \).

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

| Disjoint union: | \(|\gamma| = n \iff \gamma \in A \) then \( \mathbb{P}_C(\gamma) = \frac{x^n}{A(x)} \cdot \frac{A(x)}{C(x)} \) . . . |
| Product: | \( \mathbb{P}_C(\gamma) = \frac{x^k}{A(x)} \cdot \frac{x^{n-k}}{B(x)} = \frac{x^n}{C(x)} \). |
**Lemma (Sequences)**

Boltzmann sampler $\Gamma C$ for $\mathcal{C} = \text{SEQ}(\mathcal{A})$:

- 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: $\mathcal{C} = 1 + \mathcal{A} \mathcal{C}$ with $+, \times$ constructions.

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

Number of trials of Bernoulli RV till success is Geometric.
## Specifications with \{∪, ×, SEQ\}

<table>
<thead>
<tr>
<th>Specs</th>
<th>GF</th>
<th>Sampler</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) or (\mathcal{Z}) (atom)</td>
<td>1 or (x)</td>
<td>(\Gamma C :=) output (1) or (\bullet)</td>
</tr>
<tr>
<td>(C = A \cup B)</td>
<td>(C(x) = A(x) + B(x))</td>
<td>(\Gamma C(x) := \frac{A(x)}{C(x)} \rightarrow \Gamma B(x)</td>
</tr>
<tr>
<td>(C = A \times B)</td>
<td>(C(x) = A(x) \times B(x))</td>
<td>(\Gamma C(x) := \langle \Gamma B(x), \Gamma C(x) \rangle)</td>
</tr>
<tr>
<td>(C = SEQ(A))</td>
<td>(C(x) = \frac{1}{1 - A(x)})</td>
<td>(\Gamma C(x) :=) Geom([A(x)]) \implies \Gamma A(x))</td>
</tr>
</tbody>
</table>

*Compile sampler* from specification automatically.
Specifications with \( \{\cup, \times, \text{SEQ}\} \) — continued

**Theorem (Complexity Minithemorem)**

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

**Proof** \( \{\cup, \times, \text{SEQ}\} \): 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 \( \{\cup, \times, \text{SEQ}\} \).

Contex-free specification = recursive with \( \{\cup, \times, \text{SEQ}\} \).

**Proposition**

Regular structures *and* context-free structures have Bolzmann samplers of linear-time complexity.
Specifications with \( \{ \cup, \times, \text{SEQ} \} \) — continued (2)

**Regular specifications**

- **Binary words** with **longest run** of a’s of length < 17.
  \[
  \text{SEQ}_{<17} (\{a\}) \cdot \text{SEQ} (b \text{SEQ}_{<17} (\{a\})).
  \]

- **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**.
Contex-free specifications.

- **Binary trees**: $B = \mathbb{Z} + B \times B$.
  - Solve quadratic equation $B = x + B^2$ numerically, given $x$;
  - Output single node with probability $\frac{x}{B}$;
Else: Do two independent recursive calls to $\Gamma B(x)$.

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

Generate **Motzkin trees** [≠ 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

\[ \mathbb{P}(\gamma) \propto \frac{x^{\mid \gamma \mid}}{\mid \gamma \mid !} \implies \mathbb{P}(\gamma) = \frac{1}{C(x)} \frac{x^{\mid \gamma \mid}}{\mid \gamma \mid !} \]

\[ C(x) := \sum_n^n C_n \frac{x^n}{n!} \] 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!
Poisson law: \( P(X = k) = e^{-\lambda} \frac{\lambda^k}{k!} \).

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

Lemma

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

\[ C = \text{SET}(A) : \text{Pois}(A(x)) \implies \Gamma A(x) \]

\[ C = \text{CYC}(A) : \text{Loga}(A(x)) \implies \Gamma A(x) \]

Cf: \( C = \text{SEQ}(A) : \text{Geom}(A(x)) \implies \Gamma A(x) \).
Applies to any specifiable class of combinatorial objects

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

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

- Solve $T(x) = x e^{T(x)}$ numerically.
- Generate root ($\mathcal{Z}$);
- Choose random root degree as $\Delta := \text{Pois}(T(x))$;
- Call $\Delta$ independent copies of $\Gamma(x)$;
- Hope for the best regarding size ($\leadsto$ later)
Examples:

**Set partitions.** $\mathcal{S} = \text{SET}(\text{SET}_{\geq 1}(\mathbb{Z}))$. 

# components is $\text{Pois}(e^x - 1)$; each comp. is $\text{Pois}(x)$ $\geq 1 = \text{Vershik}.$

**Ordered set partitions.** Geometric triggers Poisson.

**Assemblies of filaments.** Poisson triggers geometric.
[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 $\mathcal{C} = \text{MSET}(\mathcal{A})$: form all finite multisets,

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

(i) **Gedanken Alg.** Scan $\mathcal{A}$ & generate $\alpha$ with multiplicity $\text{Geom}(x^{|\alpha|})$.

(ii) Observe GF equation: $C(x) = \exp(A(x)) \cdot \exp\left(\frac{1}{2} A(x^2)\right) \cdots$.

(iii) Do Poisson-controlled generator for $\mathcal{A}$ with parameter $A(x)$; repeat with $\frac{1}{2} A(x^2)$; etc.

(iv) Compute when to stop. Collect multiset.

Proof involves $\text{Geom}(\lambda) \equiv \text{Pois}(\lambda) + \text{Pois}(\frac{1}{2} \lambda^2) + \cdots$. 
Powersets and cycles

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

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

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

- **The powerset** construction $\mathcal{C} = \text{PSET}(\mathcal{A})$: 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 $\mathcal{C} = \text{MSET}_3(\mathcal{A})$: do $\Gamma A(x^3)$, etc.
Theorem (Main Complexity Theorem)

For a class $\mathcal{C}$ specified (poss. recursively) from finite sets using $+, \times, \text{SEQ}, \text{MSET}, \text{MSET}_k, \text{Cyc}, \text{Cyc}_k$,

The Boltzmann sampler $\Gamma_{\mathcal{C}}(x)$ 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
Random Generation
Boltzmann Framework
Boltzmann Samplers
Size Control and Complexity

Unions, products, and sequences
Labelled models, sets and cycles
Unlabelled sets and cycles

Functional graph

Series-parallel circuit
Complexity

- Size control

\[
\text{PGF}(\text{Size}) = \frac{C(ux)}{C(x)} \quad \implies \quad \mathbb{E}_x(\text{Size}) = \frac{xC'(x)}{C(x)}.
\]

Usually requires \( x \to \rho_C \) to get large structures.
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 $= n$. 

Analysis of size distribution of free sampler determines complexity.
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.
“Frequent” profiles: [cf Analytic Combinatorics]

Depends on singularity type of generating function.
Theorem (Complexity I)

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

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

Theorem (Complexity II)

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

Theorem (Complexity III)

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

Renewal type of algorithm at critical $\rho$. 
Size control (3): Pointing

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

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

**Uniformity at given size is preserved (only size profile is altered).**

Transforms peaked (inefficient) distributions to flat (efficient).

E.g., **binary trees** $\mathcal{B}$:

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

**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?
Some literature (all on the web!)


