A general and algorithmic method for computing the generating function of permutation classes and for their random generation

Mathilde Bouvel (LaBRI)

avec Frédérique Bassino (LIPN), Adeline Pierrot (LIAFA), Carine Pivoteau (LIGM), Dominique Rossin (LIX)

Séminaire Algo du GREYC

Guideline for the talk

Data:

- B a finite set of permutations (the excluded patterns),
- $\mathcal{C} = Av(B)$ the class of permutations that avoid every pattern of B.

Problem:

Describe an algorithm to obtain automatically from B

- some enumerative results on $\mathcal{C}$, in terms of generating function $C(z) = \sum |Av_n(B)| z^n$,
- a random sampler of permutations in $\mathcal{C}$, that is uniform on $Av_n(B)$ for each n.

Result:

Such an algorithm . . . that works under some hypothesis on $\mathcal{C}$, also tested algorithmically.

Outline

Outline

Representation of permutations

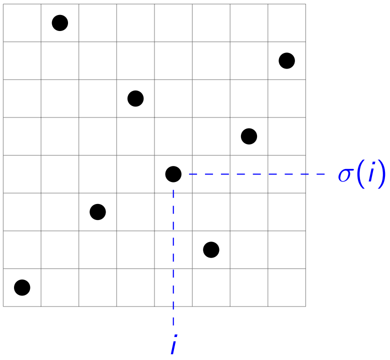
Permutation: Bijection from $[1..n]$ to itself. Set $\mathfrak{S}_n$.

- Linear representation:
  $\sigma = 1\ 8\ 3\ 6\ 4\ 2\ 5\ 7$

- Two lines representation:
  $$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 1 & 8 & 3 & 6 & 4 & 2 & 5 & 7 \end{pmatrix}$$

- Representation as a product of cycles:
  $\sigma = (1)\ (2\ 8\ 7\ 5\ 4\ 6)\ (3)$

- Graphical representation:



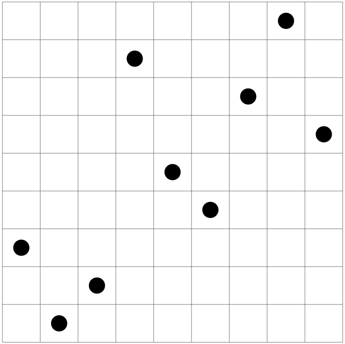
Patterns in permutations

Pattern (order) relation $\preccurlyeq$:
$\pi \in \mathfrak{S}_k$ is a pattern of $\sigma \in \mathfrak{S}_n$ if $\exists\ 1 \leq i_1 < \ldots < i_k \leq n$ such that $\sigma_{i_1} \ldots \sigma_{i_k}$ is order isomorphic ($\equiv$) to π.

Notation: $\pi \preccurlyeq \sigma$.

Equivalently:
The normalization of $\sigma_{i_1} \ldots \sigma_{i_k}$ on $[1..k]$ yields π.

Example: $2\,1\,3\,4 \preccurlyeq \mathbf{3}\,\mathbf{1}\,2\,8\,\mathbf{5}\,4\,\mathbf{7}\,9\,6$
since $3\,1\,5\,7 \equiv 2\,1\,3\,4$.



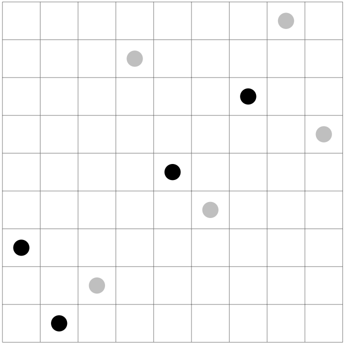
Patterns in permutations

Pattern (order) relation $\preccurlyeq$:
$\pi \in \mathfrak{S}_k$ is a pattern of $\sigma \in \mathfrak{S}_n$ if $\exists\ 1 \leq i_1 < \ldots < i_k \leq n$ such that $\sigma_{i_1} \ldots \sigma_{i_k}$ is order isomorphic ($\equiv$) to π.

Notation: $\pi \preccurlyeq \sigma$.

Equivalently:
The normalization of $\sigma_{i_1} \ldots \sigma_{i_k}$ on $[1..k]$ yields π.

Example: $2\,1\,3\,4 \preccurlyeq \mathbf{3}\,\mathbf{1}\,2\,8\,\mathbf{5}\,4\,\mathbf{7}\,9\,6$
since $3\,1\,5\,7 \equiv 2\,1\,3\,4$.

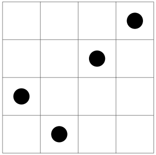


Patterns in permutations

Pattern (order) relation $\preccurlyeq$:
$\pi \in \mathfrak{S}_k$ is a pattern of $\sigma \in \mathfrak{S}_n$ if $\exists\ 1 \leq i_1 < \ldots < i_k \leq n$ such that $\sigma_{i_1} \ldots \sigma_{i_k}$ is order isomorphic ($\equiv$) to π.

Notation: $\pi \preccurlyeq \sigma$.

Equivalently:
The normalization of $\sigma_{i_1} \ldots \sigma_{i_k}$ on $[1..k]$ yields π.



Example: $2\,1\,3\,4 \preccurlyeq \mathbf{3}\,\mathbf{1}\,2\,8\,\mathbf{5}\,4\,\mathbf{7}\,9\,6$ since $3\,1\,5\,7 \equiv 2\,1\,3\,4$.

Permutation classes

Permutation class : set of permutations downward-closed for $\preccurlyeq$.

$Av(B)$: the class of permutations that avoid every pattern of B.
If B is an antichain then B is the basis of $Av(B)$.

Conversely : Every class $\mathcal{C}$ can be characterized by its basis:

$$\mathcal{C} = Av(B) \text{ for } B = \{\sigma \notin \mathcal{C} : \forall \pi \preccurlyeq \sigma \text{ such that } \pi \neq \sigma, \pi \in \mathcal{C}\}$$

A class has a unique basis.
A basis can be either finite or infinite.

Origin : [Knuth 73] with stack-sortable permutations $= Av(231)$

Enumeration[Stanley & Wilf 92][Marcus & Tardos 04] : $|\mathcal{C} \cap \mathfrak{S}_n| \leq c^n$

Problematics

- **Combinatorics**: study of classes defined by their basis.

↪ Enumeration.

↪ Exhaustive generation.

- **Algorithmics**: problematics from text algorithmics.

↪ Pattern matching, longest common pattern.

↪ Linked with testing the membership of σ to a class.

- **Combinatorics (and algorithms)**: study families of classes.

↪ A class is not always described by its basis.

↪ Obtain general results on the structure of a class...

↪ ...and do it automatically (with algorithms).

Outline

Substitution decomposition: main ideas

Analogous to the decomposition of integers as products of primes.

- [Möhring & Radermacher 84]: general framework.
- Specialization: Modular decomposition of graphs.

Relies on:

- a principle for building objects (permutations, graphs) from smaller objects: the substitution.
- some "basic objects" for this construction: **simple** permutations, **prime** graphs.

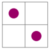
Required properties:

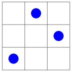
- every object can be decomposed using only "basic objects".
- this decomposition is unique.

Substitution for permutations

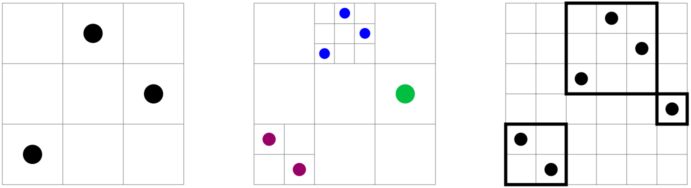
Substitution or inflation : $\sigma = \pi[\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(k)}]$.

Example : Here, $\pi = 1\,3\,2$, and

$\alpha^{(1)} = 2\,1 =$ 

$\alpha^{(2)} = 1\,3\,2 =$ 

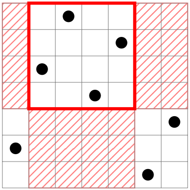
$\alpha^{(3)} = 1 =$  .



Hence $\sigma = 1\,3\,2[2\,1, 1\,3\,2, 1] = 2\,1\,4\,6\,5\,3$.

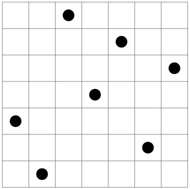
Simple permutations

Interval (or block) = set of elements of σ whose positions **and** values form intervals of integers
Example: 5 7 4 6 is an interval of 2 5 7 4 6 1 3



Simple permutation = permutation that has no interval, except the trivial intervals: $1, 2, \ldots, n$ and σ
Example: 3 1 7 4 6 2 5 is simple.

The smallest simple: 1 2, 2 1, 2 4 1 3, 3 1 4 2



Substitution decomposition of permutations

Theorem: Every σ ($\neq 1$) is uniquely decomposed as

- $12[\alpha^{(1)}, \alpha^{(2)}]$, where $\alpha^{(1)}$ is $\oplus$-indecomposable
- $21[\alpha^{(1)}, \alpha^{(2)}]$, where $\alpha^{(1)}$ is $\ominus$-indecomposable
- $\pi[\alpha^{(1)}, \ldots, \alpha^{(k)}]$, where π is simple of size $k \geq 4$

Remarks:

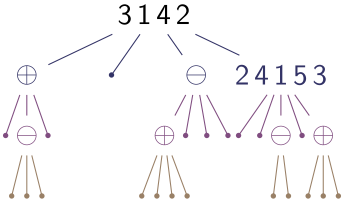
- $\oplus$-indecomposable : that cannot be written as $12[\alpha^{(1)}, \alpha^{(2)}]$
- Result stated as in [Albert & Atkinson 05]
- Can be rephrased changing the first two items into:
  - $12 \ldots k[\alpha^{(1)}, \ldots, \alpha^{(k)}]$, where the $\alpha^{(i)}$ are $\oplus$-indecomposable
  - $k \ldots 21[\alpha^{(1)}, \ldots, \alpha^{(k)}]$, where the $\alpha^{(i)}$ are $\ominus$-indecomposable

Decomposing recursively inside the $\alpha^{(i)}$ $\Rightarrow$ **decomposition tree**

Decomposition tree: witness of this decomposition

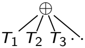
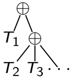
Example: Decomposition tree of $\sigma =$

10 13 12 11 14 1 18 19 20 21 17 16 15 4 8 3 2 9 5 6 7



Notations and properties:

- $\oplus = 12\ldots k$ and $\ominus = k\ldots 21$ = linear nodes.
- π simple of size ≥ 4 = prime node.
- No edge $\oplus - \oplus$ nor $\ominus - \ominus$.
- Ordered trees.

Expansion of  into  and recursively, for the version of the trees of [AA05]

$\sigma = 3\,1\,4\,2[\oplus[1, \ominus[1,1,1],1],1,\ominus[\oplus[1,1,1,1],1,1,1],2\,4\,1\,5\,3[1,1,\ominus[1,1],1,\oplus[1,1,1]]]$

Bijection between permutations and their decomposition trees.

Computation and examples of application

Computation: in linear time. [Uno & Yagiura 00] [Bui Xuan, Habib & Paul 05] [Bergeron, Chauve, Montgolfier & Raffinot 08]

In algorithms:

- Pattern matching [Bose, Buss & Lubiw 98] [Ibarra 97]
- Algorithms for bio-informatics [Bérard, Bergeron, Chauve & Paul 07] [Bérard, Chateau, Chauve, Paul & Tannier 08]

In combinatorics:

- Simple permutations [Albert, Atkinson & Klazar 03]
- Classes closed by substitution product [Atkinson & Stitt 02] [Brignall 07] [Atkinson, Ruškuc & Smith 09]
- Exhibit the structure of classes [Albert & Atkinson 05] [Brignall, Huczynska & Vatter 08a,08b] [Brignall, Ruškuc & Vatter 08]

Outline

Combinatorial classes and generating functions

Notations:

- $\mathcal{C} = \cup_{n\geq 0}\mathcal{C}_n$ with finite number $c_n = |\mathcal{C}_n|$ of objects of size n
- Generating function $C(z) = \sum c_n z^n$

Recursive description with constructors $\Rightarrow$ Equation on the g.f.:

Constructor	Notation	$C(z)$
Atom	$\mathcal{Z}$	z
Disjoint Union	$\mathcal{A}+\mathcal{B}$	$A(z)+B(z)$
Cartesian Product	$\mathcal{A}\times\mathcal{B}$	$A(z)B(z)$
Sequence	$\textsc{Seq}(\mathcal{A})$	$\dfrac{1}{1-A(z)}$
Restricted Seq.	$\textsc{Seq}_{=k}(\mathcal{A})$	$A(z)^k$

[Flajolet & Sedgewick 09]

Combinatorial classes and random samplers

Uniform sampling: objects of size n have the same probability

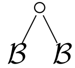
Two methods based on the recursive description of objects:

- Recursive method [Flajolet, Zimmerman & Van Cutsem 94]:
  size n chosen in advance. Requires to know the c_k for $k \leq n$.
- **Boltzmann method** [Duchon, Flajolet, Louchard & Schaeffer 04]:
  size n not fixed. Needs the evaluation of $C(z)$ at one point x.

$\mathcal{Z}$	return an atom
$\mathcal{A}+\mathcal{B}$	call $\Gamma A(x)$ with proba. $\frac{A(x)}{A(x)+B(x)}$, else $\Gamma B(x)$
$\mathcal{A}\times\mathcal{B}$	call $\Gamma A(x)$ and $\Gamma B(x)$
$\mathrm{SEQ}(\mathcal{A})$	choose k according to a geometric law of parameter $A(x)$ and call $\Gamma A(x)$ k times
$\mathrm{SEQ}_{=k}(\mathcal{A})$	call the sampler $\Gamma A(x)$ k times

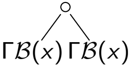
Example: binary trees

$\mathcal{B} = \cup_{n \geq 1} \mathcal{B}_n$
where $\mathcal{B}_n$ denotes the set of binary trees with n leaves.

Recursive description (also called specification): $\mathcal{B} = \bullet \quad + $ 

Equation for the g.f.: $B(z) = z + B(z)^2$, hence $B(z) = \frac{1-\sqrt{1-4z}}{2}$.

Boltzmann random sampler $\Gamma\mathcal{B}(x)$ for $\mathcal{B}$:

- Data: x, $B(x)$
- Result: a random binary tree
- Procedure:
  - Choose r uniformly at random on $[0, 1]$
  - If $\frac{x}{B(x)} < r$ then return $\bullet$
  - Else return 

Specifications for permutation classes

For all permutations, with $\mathcal{S}$ the set of all simple permutations:

$$\begin{cases} \mathfrak{S} = \bullet + \oplus(\mathfrak{S}^+, \mathfrak{S}) + \ominus(\mathfrak{S}^-, \mathfrak{S}) + \sum_{\pi \in \mathcal{S}} \pi(\mathfrak{S}, \mathfrak{S}, \dots, \mathfrak{S}) \\ \mathfrak{S}^+ = \bullet + \ominus(\mathfrak{S}^-, \mathfrak{S}) + \sum_{\pi \in \mathcal{S}} \pi(\mathfrak{S}, \mathfrak{S}, \dots, \mathfrak{S}) \\ \mathfrak{S}^- = \bullet + \oplus(\mathfrak{S}^+, \mathfrak{S}) + \sum_{\pi \in \mathcal{S}} \pi(\mathfrak{S}, \mathfrak{S}, \dots, \mathfrak{S}) \end{cases}$$

$\Rightarrow$The generating functions of $\mathfrak{S}$ and $\mathcal{S}$ are related [Albert, Atkinson & Klazar 03].

This can be adapted to (substitution-closed and arbitrary) permutation classes [Albert & Atkinson 05].

The simpler case of substitution-closed classes

A permutation class $\mathcal{C}$ is **substitution-closed** when $\pi[\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(k)}] \in \mathcal{C}$ for all $\pi, \alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(k)} \in \mathcal{C}$.

Hence, with $\mathcal{S}_{\mathcal{C}} = \mathcal{C} \cap \mathcal{S}$ the set of simple permutations in $\mathcal{C}$:

$$\begin{cases} \mathcal{C} = \bullet + \oplus[\mathcal{C}^{+}, \mathcal{C}] + \ominus[\mathcal{C}^{-}, \mathcal{C}] + \sum_{\pi \in \mathcal{S}_{\mathcal{C}}} \pi[\mathcal{C}, \mathcal{C}, \ldots, \mathcal{C}] \\ \ldots \end{cases}$$

When $\mathcal{S}_{\mathcal{C}}$ is finite, this is a simple family of trees in the sense of [Flajolet & Sedgewick 09].

$\Rightarrow$Enumerative results and random samplers can be obtained by efficient algorithms.

For general permutation classes

For non substitution-closed classes, we have only a strict inclusion:

$$\mathcal{C} \subsetneq \bullet + \oplus[\mathcal{C}^+, \mathcal{C}] + \ominus[\mathcal{C}^-, \mathcal{C}] + \sum_{\pi \in \mathcal{S}_{\mathcal{C}}} \pi[\mathcal{C}, \mathcal{C}, \cdots, \mathcal{C}]$$

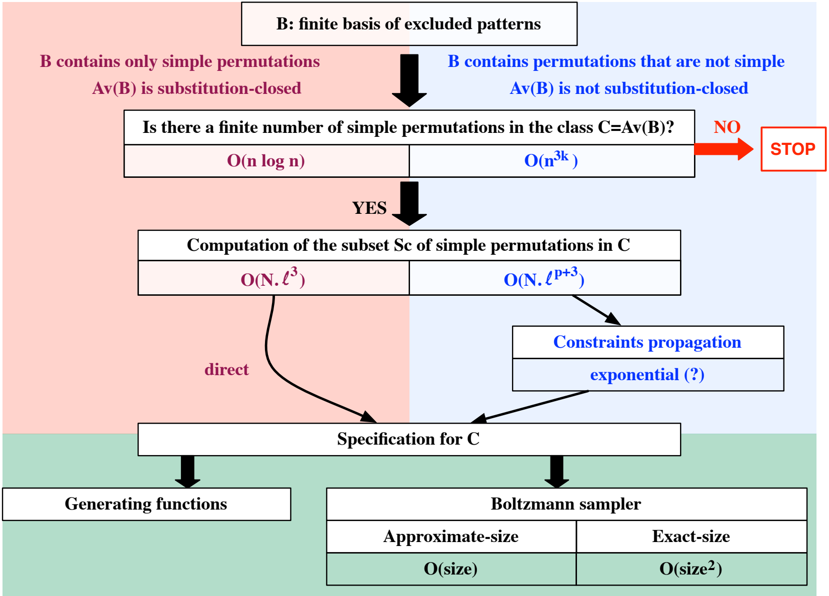
Example: $\ominus[12, 1] \notin Av(231)$ whereas $12, 1 \in Av(231)$.

The system describing $\mathcal{C}$ has to be refined with new equations for these constraints. The system can be computed by an algorithm.

$\Rightarrow$Enumerative results and random samplers can be obtained algorithmically, but this is less efficient.

Outline

Summary of results



First semi-decision procedure

Theorem [Albert & Atkinson 05]: If $\mathcal{C}$ contains a finite number of simple permutations, then $\mathcal{C}$ has a finite basis and an algebraic g.f..

Constructive proof: compute, for each given class,

- the specification for decomposition trees of $\mathcal{C}$
- a system of equations satisfied by the g.f.

from the finite set of simple permutations in $\mathcal{C}$

Testing the precondition:

- Semi-decision procedure

$\hookrightarrow$ Find simples of size $4, 5, 6, \ldots$ until k and $k+1$ for which there are 0 simples [Schmerl & Trotter 93]

- "Very exponential" ($\sim n!$) computation of the simples in $\mathcal{C}$

Step 1: Is there a finite number of simple permutations in $\mathcal{C}$? A first decision result

Theorem [Brignall, Ruškuc & Vatter 08]: It is decidable whether $\mathcal{C}$ given by its finite basis contains a finite number of simples.

Prop: $\mathcal{C} = Av(B)$ contains infinitely many simples iff $\mathcal{C}$ contains:

1. either infinitely many parallel permutations
2. or infinitely many simple wedge permutations
3. or infinitely many proper pin-permutations

	Decision procedure	Complexity
1. and 2. :	pattern matching of patterns of size 3 or 4 in the $\beta \in B$.	Polynomial $\mathcal{O}(n \log n)$
3. :	Decidability with automata techniques	Decidable 2ExpTime

Polynomial algorithms for the finite number of simples

Points similar to [BRV 08] :

- Encoding by pin words on $\{1, 2, 3, 4, L, R, U, D\}$
- Construction of automata

Study of pin-permutations [BBR 09] $\Rightarrow$ better understanding of the relationship between pin words and patterns in permutations

Points specific to [BBPR 10 & 11] :

- Polynomial construction of a **(deterministic, complete)** automaton for the language $\mathcal{L}$ = pin words of proper pin-permutations containing some $\beta \in B$
- Is this language co-finite ? Polynomial.

$\hookrightarrow$ Yes iff the class contains finitely many simples.

Polynomial w.r.t. $n = \sum_{\beta \in B} |\beta|$, but $k = |B|$ is an exponent.

Improvements for substitution-closed classes

Prop: $\mathcal{C} = Av(B)$ is substitution-closed iff B contains only simple permutations.

For simple β, $\beta \preccurlyeq \pi$ translates into a factor relation on pin words.
$\Rightarrow$ B gives a set of factors F (whose lengths sum to $\mathcal{O}(n)$) such that w has a factor in F iff $\beta \preccurlyeq \pi_w$ for some $\beta \in B$

[Aho & Corasick 75]:
build in linear time a complete deterministic automaton $\mathcal{A}_F$ recognizing the language of words containing a factor F

$\mathcal{L}(\mathcal{A}_F)$ co-finite iff finite number of simples in $\mathcal{C}$
. . . and testing the co-finiteness of $\mathcal{L}(\mathcal{A}_F)$ is in linear time.

Theorem: Testing the finiteness of the number of simple permutations in a substitution-closed class is solved in $\mathcal{O}(n \log n)$

Polynomial algorithm for general classes

When β is not simple (but is a pin permutation), $\beta \preccurlyeq \pi$ translates into a piecewise factor relation on pin words.

Def: $f = (f_1, f_2, \ldots, f_k)$ is a piecewise factor of w iff $w = w_0 f_1 w_1 f_2 w_2 \ldots w_{k-1} f_k w_k$.

Piecewise factors F_β corresponding to $\beta \in B$ are computed inductively on the decomposition trees of β.
And similarly for the deterministic automaton $\mathcal{A}_\beta$ recognizing the language of words containing a piecewise factor in F_β.

Construction of $\mathcal{A}_\beta$ in $\mathcal{O}(|\beta|^3)$.
Then build the product of the $\mathcal{A}_\beta$ for $\beta \in B$ (deterministic union).

Theorem: Testing the finiteness of the number of simple permutations in a permutation class is solved in $\mathcal{O}(n^{3k})$

Step 2: Finding the set of simple permutations in $\mathcal{C}$

Starting point: Find simple permutations in $\mathcal{C}$ of size $4, 5, 6, \ldots$ until k and $k+1$ for which there are 0 simples

Problem: There are $\sim \frac{n!}{e^2}$ simple permutations of size n

Reduce the number of simples σ of size n that are candidate to the membership to $\mathcal{C}$ [Pierrot & Rossin, 11].

Prop: The simples of $\mathcal{C}_{n+1}$ can be described as one-point (or special two-points) extensions of the simples of $\mathcal{C}_n$
$\Rightarrow$ There are at most $\mathcal{O}(n^2.|\mathcal{S} \cap \mathcal{C}_n|)$ candidates of size $n+1$.

Test whether σ contains an occurrence of $\beta \in B$: in $\mathcal{O}(n^{|\beta|})$.

Theorem: Computing the finite set of simple permutations in $\mathcal{C}$ is done in $\mathcal{O}(N \cdot \ell^{p+3})$ with $N = |\mathcal{S} \cap \mathcal{C}|$, $p = \max\{|\beta| : \beta \in B\}$ and $\ell = \max\{|\pi| : \pi \in \mathcal{S} \cap \mathcal{C}\}$

Refinement for substitution-closed classes

Prop: $\mathcal{C} = Av(B)$ is substitution-closed iff B contains only simples.

Prop [Pierrot & Rossin, 11]: If $\beta \preccurlyeq \sigma$ for β and σ simples, then there are simples $\beta = \sigma_1 \preccurlyeq \sigma_2 \ldots \preccurlyeq \sigma_k = \sigma$ s.t. for all i, $|\sigma_i| - |\sigma_{i-1}| = 1$ (or 2 in special cases).

Improvement of the complexity:

- Avoid testing occurrences of $\beta \in B$ in σ candidate simple of $\mathcal{C}$.
- Instead, test whether for every one point (or special two points) deletion in σ resulting in σ' simple, then $\sigma' \in \mathcal{C}$.

$\Rightarrow$ It is more efficient for computing $\mathcal{S} \cap \mathcal{C}_{n+1}$ from $\mathcal{S} \cap \mathcal{C}_n$.

Theorem: Computing the finite set of simple permutations in $\mathcal{C}$ is done in $\mathcal{O}(N \cdot \ell^5)$ for substitution-closed classes.

Step 3: Compute the specification for $\mathcal{C}$

From the set $\mathcal{S}_\mathcal{C}$ of simple permutations in $\mathcal{C}$, the specification for the substitution closure $\hat{\mathcal{C}}$ of $\mathcal{C}$ is obtained immediately:

$$\begin{cases} \hat{\mathcal{C}} = \bullet + \oplus[\hat{\mathcal{C}}^+, \hat{\mathcal{C}}] + \ominus[\hat{\mathcal{C}}^-, \hat{\mathcal{C}}] + \sum_{\pi \in \mathcal{S}_\mathcal{C}} \pi[\hat{\mathcal{C}}, \hat{\mathcal{C}}, \dots, \hat{\mathcal{C}}] \\ \hat{\mathcal{C}}^+ = \bullet + \ominus[\hat{\mathcal{C}}^-, \hat{\mathcal{C}}] + \sum_{\pi \in \mathcal{S}_\mathcal{C}} \pi[\hat{\mathcal{C}}, \hat{\mathcal{C}}, \dots, \hat{\mathcal{C}}] \\ \hat{\mathcal{C}}^- = \bullet + \oplus[\hat{\mathcal{C}}^+, \hat{\mathcal{C}}] + \sum_{\pi \in \mathcal{S}_\mathcal{C}} \pi[\hat{\mathcal{C}}, \hat{\mathcal{C}}, \dots, \hat{\mathcal{C}}] \end{cases}$$

- If $\mathcal{C}$ is substitution-closed, $\mathcal{C} = \hat{\mathcal{C}}$ and we are done.
- Otherwise, $\mathcal{C} = \hat{\mathcal{C}}\langle B^\star \rangle$ and propagate the constraints from $B^\star = \{\beta \in B \ : \ \beta \text{ is not simple } \}$ into the subtrees.

Step 3: Compute the specification for $\mathcal{C}$

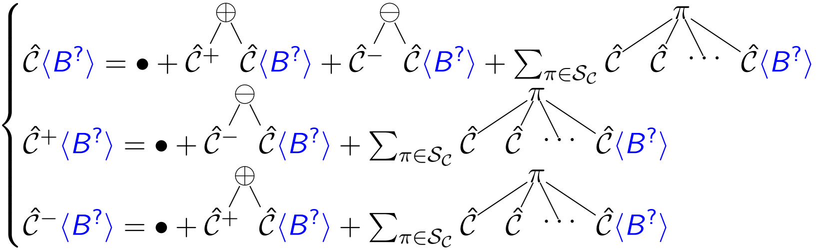
From the set $\mathcal{S}_{\mathcal{C}}$ of simple permutations in $\mathcal{C}$, the specification for the substitution closure $\hat{\mathcal{C}}$ of $\mathcal{C}$ is obtained immediately:

$$\begin{cases} \hat{\mathcal{C}}\langle B^\star\rangle = \bullet + \oplus[\hat{\mathcal{C}}^+, \hat{\mathcal{C}}\langle B^\star\rangle] + \ominus[\hat{\mathcal{C}}^-, \hat{\mathcal{C}}\langle B^\star\rangle] + \sum_{\pi\in\mathcal{S}_{\mathcal{C}}} \pi[\hat{\mathcal{C}}, \hat{\mathcal{C}}, \dots, \hat{\mathcal{C}}\langle B^\star\rangle] \\ \hat{\mathcal{C}}^+\langle B^?\rangle = \bullet + \ominus[\hat{\mathcal{C}}^-, \hat{\mathcal{C}}\langle B^?\rangle] + \sum_{\pi\in\mathcal{S}_{\mathcal{C}}} \pi[\hat{\mathcal{C}}, \hat{\mathcal{C}}, \dots, \hat{\mathcal{C}}\langle B^?\rangle] \\ \hat{\mathcal{C}}^-\langle B^?\rangle = \bullet + \oplus[\hat{\mathcal{C}}^+, \hat{\mathcal{C}}\langle B^?\rangle] + \sum_{\pi\in\mathcal{S}_{\mathcal{C}}} \pi[\hat{\mathcal{C}}, \hat{\mathcal{C}}, \dots, \hat{\mathcal{C}}\langle B^?\rangle] \end{cases}$$

- If $\mathcal{C}$ is substitution-closed, $\mathcal{C} = \hat{\mathcal{C}}$ and we are done.
- Otherwise, $\mathcal{C} = \hat{\mathcal{C}}\langle B^\star\rangle$ and propagate the constraints from $B^\star = \{\beta \in B \ : \ \beta \text{ is not simple }\}$ into the subtrees.

Step 3: Compute the specification for $\mathcal{C}$

From the set $\mathcal{S}_{\mathcal{C}}$ of simple permutations in $\mathcal{C}$, the specification for the substitution closure $\hat{\mathcal{C}}$ of $\mathcal{C}$ is obtained immediately:



- If $\mathcal{C}$ is substitution-closed, $\mathcal{C} = \hat{\mathcal{C}}$ and we are done.
- Otherwise, $\mathcal{C} = \hat{\mathcal{C}}\langle B^\star \rangle$ and propagate the constraints from $B^\star = \{\beta \in B \ : \ \beta \text{ is not simple } \}$ into the subtrees.

Constraint propagation 1/2

Embeddings of $\beta \in B^\star$ into $\pi \in \mathcal{S}_\mathcal{C}$

- Example: for $\ominus[\mathcal{C}^-, \mathcal{C}]\langle 231 \rangle$, and for the embedding $(23, 1) \hookrightarrow (2, 1)$, we get $\mathcal{C}^-\langle 12 \rangle$.
- additional restrictions α in $B^?$ that are blocks of $\beta \in B^\star$
- and do it inductively while new constraints α appear
- this terminates since each $\alpha \preccurlyeq \beta$ for some $\beta \in B^\star$

Constraint propagation 1/2

Embeddings of $\beta \in B^\star$ into $\pi \in \mathcal{S}_\mathcal{C}$

- Example: for $\ominus[\mathcal{C}^-, \mathcal{C}]\langle 231 \rangle$, and for the embedding $(23, 1) \hookrightarrow (2, 1)$, we get $\mathcal{C}^-\langle 12 \rangle$.
- additional restrictions α in $B^?$ that are blocks of $\beta \in B^\star$
- and do it inductively while new constraints α appear
- this terminates since each $\alpha \preccurlyeq \beta$ for some $\beta \in B^\star$

Result: A system describing $\mathcal{C}$, that may be ambiguous
Example: For $2413[\mathcal{C}, \mathcal{C}, \mathcal{C}, \mathcal{C}]\langle 1234 \rangle$,
the embeddings $(1, 234) \hookrightarrow (2, 4)$ and $(1, 234) \hookrightarrow (1, 3)$
produce the terms $2413[\mathcal{C}, \mathcal{C}\langle 123 \rangle, \mathcal{C}, \mathcal{C}]$ and $2413[\mathcal{C}, \mathcal{C}, \mathcal{C}, \mathcal{C}\langle 123 \rangle]$
whose intersection is not empty.

Constraint propagation 2/2

Disambiguation of the system:

- Use formulas of the type $A \cup B = A \cap B \uplus \overline{A} \cap B \uplus A \cap \overline{B}$
- In complement set, excluded patterns become mandatory patterns: $\mathcal{C}_\gamma$ for $\gamma \preccurlyeq \beta \in B^\star$
- Propagate also mandatory restrictions

Constraint propagation 2/2

Disambiguation of the system:

- Use formulas of the type $A \cup B = A \cap B \ \uplus \ \overline{A} \cap B \ \uplus \ A \cap \overline{B}$
- In complement set, excluded patterns become mandatory patterns: $\mathcal{C}_\gamma$ for $\gamma \preccurlyeq \beta \in B^\star$
- Propagate also mandatory restrictions

Result: An unambiguous system describing $\mathcal{C}$, where the left-hand-sides are $\mathcal{C}^{\epsilon}_{\gamma_1,\ldots,\gamma_p}\langle\alpha_1,\ldots,\alpha_k\rangle$ with $\epsilon \in \{\ \ ,+,-\}$.

Termination: all α_i and γ_j are patterns of some $\beta \in B^\star$

Constraint propagation 2/2

Disambiguation of the system:

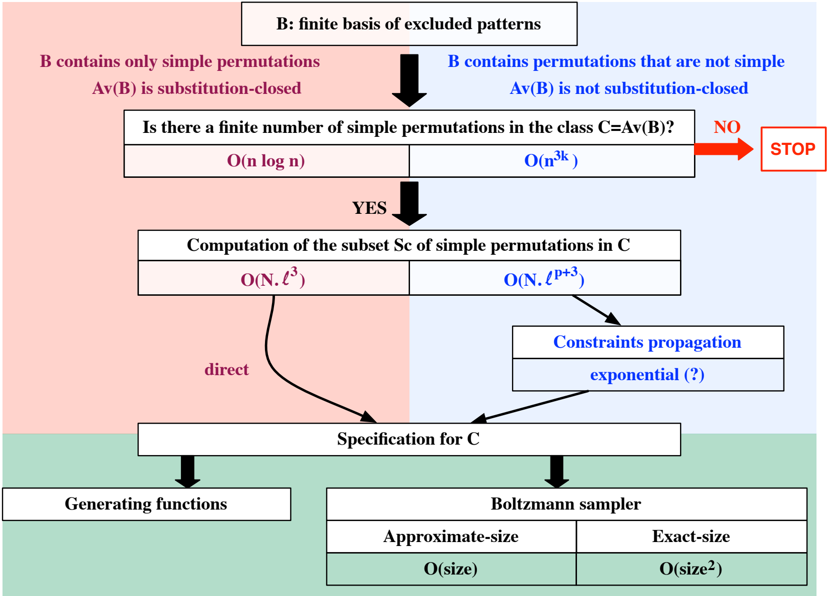
- Use formulas of the type $A \cup B = A \cap B \uplus \overline{A} \cap B \uplus A \cap \overline{B}$
- In complement set, excluded patterns become mandatory patterns: $\mathcal{C}_\gamma$ for $\gamma \preccurlyeq \beta \in B^\star$
- Propagate also mandatory restrictions

Result: An unambiguous system describing $\mathcal{C}$, where the left-hand-sides are $\mathcal{C}^{\epsilon}_{\gamma_1,\ldots,\gamma_p}\langle \alpha_1, \ldots, \alpha_k \rangle$ with $\epsilon \in \{\ ,+,-\}$.

Termination: all α_i and γ_j are patterns of some $\beta \in B^\star$

Theorem: The propagation of the constraints to obtain a specification for $\mathcal{C}$ is algorithmic, but there is an explosion of the number of equations in the system.

Putting things together



Outline

What next?

About the algorithm:

- Implementation in progress
- Complexity analysis of step 3 (explosion of the system)
- Dependency of the complexity of Boltzmann random samplers w.r.t. the size of the specification

With the algorithm:

- From the specifications, estimate growth rates of classes
- Are random permutations in $\mathcal{C}$ "like" in $\mathfrak{S}$?
- Compare statistics on $\mathcal{C}$ and $\mathfrak{S}$, or on $\mathcal{C}_1$ and $\mathcal{C}_2$

Related questions:

- How general is our algorithm?
- Classes with infinite set of simples, but finitely described?
- Use specification of a class to decide membership efficiently?