Some algorithmic and combinatorial problems on permutation classes

The point of view of decomposition trees

Mathilde Bouvel

PhD Defense, 2009 December the 4th
Outline

1. Objects studied: Permutations, Patterns and Classes
2. Main tool: decomposition trees
3. Applications in algorithmics
4. Structure of permutations classes in combinatorics
5. A transverse example: perfect sorting by reversals
6. Conclusion and perspectives
Outline

1. Objects studied: Permutations, Patterns and Classes
2. Main tool: decomposition trees
3. Applications in algorithmics
4. Structure of permutations classes in combinatorics
5. A transverse example: perfect sorting by reversals
6. Conclusion and perspectives
Permutation: Bijection from \([1..n]\) to itself. Set \(S_n\).

- **Linear representation**:
  \[
  \sigma = 1 \ 8 \ 3 \ 6 \ 4 \ 2 \ 5 \ 7
  \]

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

- **Graphical representation**:
  \[
  \sigma(i) \]

- **Representation as a product of cycles**:
  \[
  \sigma = (1) \ (2 \ 8 \ 7 \ 5 \ 4 \ 6) \ (3)
  \]
Patterns in permutations

**Pattern (order) relation** \( \preceq \):

\[ \pi \in S_k \text{ is a pattern of } \sigma \in S_n \text{ if } \exists \ 1 \leq i_1 < \ldots < i_k \leq n \text{ such that } \sigma_{i_1} \ldots \sigma_{i_k} \text{ is order isomorphic (} \equiv \text{) to } \pi. \]

Notation: \( \pi \preceq \sigma \).

**Equivalently**:
The normalization of \( \sigma_{i_1} \ldots \sigma_{i_k} \) on \([1..k]\) yields \( \pi \).

**Example**: \( 2 \ 1 \ 3 \ 4 \preceq 3 \ 1 \ 2 \ 8 \ 5 \ 4 \ 7 \ 9 \ 6 \) since \( 3 \ 1 \ 5 \ 7 \equiv 2 \ 1 \ 3 \ 4 \).
Patterns in permutations

**Pattern (order) relation** $\preceq$:

$\pi \in S_k$ is a pattern of $\sigma \in S_n$ 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 $\pi$.

Notation: $\pi \preceq \sigma$.

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

**Example** : $2\;1\;3\;4 \preceq 3\;1\;2\;8\;5\;4\;7\;9\;6$
since $3\;1\;5\;7 \equiv 2\;1\;3\;4$. 
Patterns in permutations

**Pattern (order) relation** \(\preceq\) : 

\(\pi \in S_k\) is a pattern of \(\sigma \in S_n\) 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 \(\pi\).

Notation: \(\pi \preceq \sigma\).

**Equivalently**: 
The normalization of \(\sigma_{i_1} \ldots \sigma_{i_k}\) on \([1..k]\) yields \(\pi\).

**Example**: \(2 \ 1 \ 3 \ 4 \preceq 3 \ 1 \ 2 \ 8 \ 5 \ 4 \ 7 \ 9 \ 6\) since \(3 \ 1 \ 5 \ 7 \equiv 2 \ 1 \ 3 \ 4\).
Permutation classes

**Permutation class**: set of permutations downward-closed for \(<\).

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

Conversely : Every class \(C\) can be characterized by its basis :

\[
C = S(B) \text{ for } B = \{\sigma \notin C : \forall \pi \preceq \sigma \text{ such that } \pi \neq \sigma, \pi \in C\}
\]

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

Origine : [Knuth 73] with stack-sortable permutations = \(S(231)\)

Enumeration[Stanley & Wilf 92][Marcus & Tardos 04] :
\[
|C \cap S_n| \leq c^n
\]
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 $\sigma$ to a class.

- **Combinatorics (and algorithms)**: studying classes as a whole.
  - A class is not always described by its basis.
  - Detect automatically the structure of a class.
Outline

1. Objects studied: Permutations, Patterns and Classes
2. Main tool: decomposition trees
3. Applications in algorithmics
4. Structure of permutations classes in combinatorics
5. A transverse example: perfect sorting by reversals
6. Conclusion and perspectives
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

\[
\begin{align*}
\alpha^{(1)} &= 2\ 1 = \\
\alpha^{(2)} &= 1\ 3\ 2 = \\
\alpha^{(3)} &= 1 = \\
\end{align*}
\]

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 $\sigma$ 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 $\sigma$

*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 $\sigma (\neq 1)$ is uniquely decomposed as

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

**Remarks**:

- $\oplus$-indecomposable: that cannot be written as $12[\alpha^{(1)}, \alpha^{(2)}]$
- Rephrasing a result of [Albert & Atkinson 05]
- The $\alpha^{(i)}$ are the maximal strong intervals of $\sigma$

Decomposing recursively inside the $\alpha^{(i)} \Rightarrow$ decomposition tree
Main tool: decomposition trees

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.
- $\pi$ simple of size $\geq 4$ = prime node.
- No edge $\oplus - \oplus$ nor $\ominus - \ominus$.
- Ordered trees.

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

1. Objects studied: Permutations, Patterns, and Classes
2. Main tool: decomposition trees
3. Applications in algorithmics
4. Structure of permutations classes in combinatorics
5. A transverse example: perfect sorting by reversals
6. Conclusion and perspectives
Pattern matching

Problem, which is \( NP \)-hard:

- **Input**: pattern \( \sigma \) (size \( k \)), permutation \( \tau \) (size \( n \)).
- **Output**: an occurrence of \( \sigma \) in \( \tau \) if it exists.

**Restriction**: \( \sigma \) is separable. **Polynomial** subproblem.

**Separable** permutations:
- **Definition by excluded patterns**: \( S(2413, 3142) \)
- **Other definition**: having a separating tree
- **Characterization**: decomposition tree with no prime node
Dynamic Programming [Bose, Buss & Lubiw 98] [Ibarra 97]

- following the guide = separating tree of $\sigma$
- from the leaves to the root
- for windows of positions and values

Complexity:  

[Bose, Buss & Lubiw 98]  
- $O(kn^6)$ in time  
- $O(kn^4)$ in space

[Ibarra 97]  
- $O(kn^4)$ in time  
- $O(kn^3)$ in space

$\Rightarrow$ Polynomial
Generalization with decomposition trees

Method:

- Dynamic programming.
- Consider further the prime nodes of decomposition trees.

Solutions obtained: [B. & Rossin 06] [B., Rossin & Vialette 07]

- Pattern matching of any pattern in $O(kn^{2d+2})$
- Finding a longest common pattern between two permutations, one of which is separable, in $O(\min(n_1, n_2)n_1n_2^6)$
- Finding a longest common pattern between two permutations in $O(\min(n_1, n_2)n_1n_2^{2d_1+2})$

with $d =$ maximal arity of a prime node
## Outline

1. Objects studied: Permutations, Patterns and Classes
2. Main tool: decomposition trees
3. Applications in algorithmics
4. Structure of permutations classes in combinatorics
5. A transverse example: perfect sorting by reversals
6. Conclusion and perspectives
Structure in permutation classes

**Theorem** [Albert & Atkinson 05] : If \( C \) contains a **finite** number of simple permutations, then

- \( C \) has a **finite basis**
- \( C \) has an **algebraic** generating function \((= \sum_n |C \cap S_n| x^n)\)

**Proof** : relies on the substitution decomposition.
**Construction** : compute the generating function from the simples in \( C \)

**Algorithmically** :
- **Semi-decision** procedure
- \( \rightarrow \) Find simples of size 4, 5, 6, \ldots until \( k \) and \( k + 1 \) for which there are **0** simples [Schmerl & Trotter 93]
- **“Very exponential”** (~ \( n! \)) computation of the simples in \( C \)
Finite number of simple permutations : decision

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

**Prop.** $\mathcal{C} = S(B)$ 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**

<table>
<thead>
<tr>
<th>Decision procedure</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. and 2. : pattern matching of patterns of size 3 or 4 in the $\beta \in B$.</td>
<td>Polynomial</td>
</tr>
<tr>
<td>3. : Decidability with automata techniques</td>
<td>Decidable 2ExpTime</td>
</tr>
</tbody>
</table>
The class of pin-permutations

Pin-permutation = that admits a **pin representation**, i.e. a sequence \((p_1, \ldots, p_n)\) where each \(p_i\) satisfies:

- the exteriority condition
- and
- either the separation condition
- or the independence condition

\[
\text{Bounding box of } \{p_1, \ldots, p_{i-1}\}
\]

Encoding by **pin words** on \(\{1, 2, 3, 4, L, R, U, D\}\) with \(\frac{2}{3} \frac{1}{4}\)
The class of pin-permutations

Pin-permutation = that admits a pin representation, i.e. a sequence \((p_1, \ldots, p_n)\) where each \(p_i\) satisfies:

- the exteriority condition
- and
- either the separation condition
- or the independence condition

\[
\text{bounding box of } \{p_1, \ldots, p_{i-1}\}
\]

Encoding by pin words on \(\{1, 2, 3, 4, L, R, U, D\}\) with \(\frac{2}{3} \mid \frac{1}{4}\)
The class of pin-permutations

Pin-permutation = that admits a pin representation, i.e. a sequence \((p_1, \ldots, p_n)\) where each \(p_i\) satisfies:

- the exteriority condition
- and
- either the separation condition
- or the independence condition

\[\text{bounding box of } \{p_1, \ldots, p_{i-1}\}\]

Encoding by pin words on \(\{1, 2, 3, 4, L, R, U, D\}\) with \(\frac{2}{3}, \frac{1}{4}\).
The class of pin-permutations

Pin-permutation = that admits a pin representation, i.e. a sequence \((p_1, \ldots, p_n)\) where each \(p_i\) satisfies:

- the exteriority condition
- and
- either the separation condition
- or the independence condition

\[
\text{Example:} \quad \begin{array}{c}
p_1 \downarrow \quad p_2 \quad p_3 \quad p_4 \\
\{p_1, \ldots, p_i-1\} \quad \text{bounding box}
\end{array}
\]

Encoding by pin words on \(\{1, 2, 3, 4, L, R, U, D\}\) with \(\frac{2}{3} \frac{1}{4}\)
The class of pin-permutations

Pin-permutation = that admits a pin representation, i.e. a sequence \((p_1, \ldots, p_n)\) where each \(p_i\) satisfies:

- the exteriority condition
- and
- either the separation condition
- or the independence condition

\[= \text{bounding box of } \{p_1, \ldots, p_{i-1}\}\]

Encoding by pin words on \(\{1, 2, 3, 4, L, R, U, D\}\) with \(\frac{2}{3} | \frac{1}{4}\)
The class of pin-permutations

Pin-permutation = that admits a pin representation, i.e. a sequence \((p_1, \ldots, p_n)\) where each \(p_i\) satisfies:

- the exteriority condition

- and

- either the separation condition

- or the independence condition

\[ \text{bounding box of } \{p_1, \ldots, p_{i-1}\} \]

Example:

Encoding by pin words on \(\{1, 2, 3, 4, L, R, U, D\}\) with \(\frac{2}{3} | \frac{1}{4}\)
The class of pin-permutations

Pin-permutation = that admits a pin representation, i.e. a sequence \((p_1, \ldots, p_n)\) where each \(p_i\) satisfies:

- the exteriority condition
- and
- either the separation condition
- or the independence condition

\[= \text{bounding box of } \{p_1, \ldots, p_{i-1}\}\]

Encoding by pin words on \(\{1, 2, 3, 4, L, R, U, D\}\) with \(\frac{2}{3} | \frac{1}{4}\)

Example:

\begin{align*}
\text{1 U R D 3} & \\
p_1 & \\
p_2 & \\
p_3 & \\
p_4 & \\
p_5 & \\
p_6 &
\end{align*}
The class of pin-permutations

Pin-permutation = that admits a pin representation, i.e. a sequence \((p_1, \ldots, p_n)\) where each \(p_i\) satisfies:

- the exteriority condition
- and
- either the separation condition
- or the independence condition

\[ = \text{bounding box of } \{p_1, \ldots, p_{i-1}\} \]

Example:

![Diagram of pin-permutations](image)

Encoding by pin words on \(\{1, 2, 3, 4, L, R, U, D\}\) with \(\frac{2}{3} \frac{1}{4}\)
The class of pin-permutations

Pin-permutation = that admits a pin representation, i.e. a sequence $(p_1, \ldots, p_n)$ where each $p_i$ satisfies:

- the exteriority condition
- and
- either the separation condition
- or the independence condition

$\text{bounding box of } \{p_1, \ldots, p_{i-1}\}$

Encoding by pin words on \{1, 2, 3, 4, L, R, U, D\} with $\frac{2}{3}, \frac{1}{4}$

Mathilde Bouvel

Some algorithmic and combinatorial problems on permutation classes
The class of pin-permutations

Pin-permutation = that admits a pin representation, i.e. a sequence \((p_1, \ldots, p_n)\) where each \(p_i\) satisfies:

- the exteriority condition
- and
- either the separation condition
- or the independence condition

\[
\text{bounding box of } \{p_1, \ldots, p_{i-1}\}
\]

Encoding by pin words on \(\{1, 2, 3, 4, L, R, U, D\}\) with \(\frac{2}{3} | \frac{1}{4}\)

Example:

\[
\begin{array}{c}
p_7 \quad p_8 \\
p_3 \quad p_4 \\
p_1 \quad p_5 \\
p_6 \\
1 \ U \ R \ D \ 3 \ U \ R
\end{array}
\]
Some results on pin-permutations (1/2)

- Characterization of their decomposition trees [Bassino, B. & Rossin 09]

\[ \mathcal{P} = \bullet + \]

\[ \mathcal{E}^+ \mathcal{E}^+ \ldots \mathcal{E}^+ + \mathcal{E}^+ \ldots \mathcal{E}^+ \]

\[ \mathcal{N}^+(\mathcal{P}) \]

\[ \mathcal{E}^- \mathcal{E}^- \ldots \mathcal{E}^- + \mathcal{E}^- \ldots \mathcal{E}^- \]

\[ \mathcal{N}^-(\mathcal{P}) \]

\[ + \]

\[ \alpha \]

\[ \mathcal{P}\backslash\{\bullet\} \]

\[ + \]

\[ \beta^+ \]

\[ \mathcal{P}\backslash\{\bullet\} \]

\[ + \]

\[ \beta^- \]

\[ \mathcal{P}\backslash\{\bullet\} \]
Some results on pin-permutations (2/2)

- **Computation of the generating function**: rational [BBR09]
  \[ P(z) = z \frac{8z^6 - 20z^5 - 4z^4 + 12z^3 - 9z^2 + 6z - 1}{8z^8 - 20z^7 + 8z^6 + 12z^5 - 14z^4 + 26z^3 - 19z^2 + 8z - 1} \]

- **Infinite** basis (still to be determined) [BBR09]

- **Polynomial** algorithm checking whether the number of simples in \( S(B) \) is finite [Bassino, B., Pierrot & Rossin], instead of the decision procedure of [BRV08]
Polynomial algorithm for the finite number of simples

Points similar to [BRV08]:
- Encoding by pin words on \{1, 2, 3, 4, L, R, U, D\}
- Construction of automata

Study of pin-permutations ⇒ better understanding of the relationship between pin words and patterns in permutations

Points specific to [BBPR]:
- **Polynomial** construction of a (deterministic, complete) automaton for the language \( \mathcal{L} = \text{pin words of proper pin-permutations containing some } \beta \in B \)
- Is this language co-finite? **Polynomial**.
  → Yes iff the class contains finitely many simples.
Automatic computation of the generating function

What is done:
- Deciding the finite number of simples
  - Polynomial
- Computing the simples in the class
  - Exponential
- Computing the (algebraic) generating function from the simples
  - Possible on any example

What remains to do:
- **Automatically** compute the generating function from the simples
- **Polynomial** computation of the set of simples in a class
- If $C$ is not given by its finite basis?
Outline

1. Objects studied: Permutations, Patterns and Classes
2. Main tool: decomposition trees
3. Applications in algorithmics
4. Structure of permutations classes in combinatorics
5. A transverse example: perfect sorting by reversals
6. Conclusion and perspectives
A transverse example: perfect sorting by reversals

Motivations and the model

- Genomes = sequences of genes
- Only one type of mutation is possible
- Goal: evolution scenario
- Group of common genes

≈ Signed permutations
≈ Reversal = reversing a window while changing the signs
≈ Sequence of reversals
≈ Interval of permutations

Mathilde Bouvel
Some algorithmic and combinatorial problems on permutation classes
A transverse example: perfect sorting by reversals

Perfect sorting by reversals

- **Input**: Two signed permutations $\sigma_1$ and $\sigma_2$
- **Output**: A parcimonious **perfect** scenario from $\sigma_1$ to $\sigma_2$ or $\overline{\sigma_2}$

We can always assume that $\sigma_2 = Id = 1 \ 2 \ldots \ n$

**Sorting by reversals**: polynomial [Hannenhalli & Pevzner 99]

**Perfect** sorting by reversals:

- **NP-hard** problem [Figeac & Varré 04]
- **FPT algorithm** [Bérard, Bergeron, Chauve & Paul 07]: uses the decomposition tree, in time $O(2^p \cdot n^{O(1)})$
- Complexity parametrized by $p =$ number of prime nodes (with a prime parent)
A transverse example: perfect sorting by reversals

Idea of the algorithm on an example

$$\sigma_1 = 5 \ 6 \ 7 \ 9 \ 4 \ 3 \ 1 \ 2 \ 8 \ 10 \ 17 \ 13 \ 15 \ 12 \ 11 \ 14 \ 18 \ 19 \ 16$$
A transverse example: perfect sorting by reversals

Idea of the algorithm on an example

$\sigma_1 = 5 \ 6 \ 7 \ 9 \ 4 \ 3 \ 1 \ 2 \ 8 \ 10 \ 17 \ 13 \ 15 \ 12 \ 11 \ 14 \ 18 \ 19 \ 16$

Leaves: sign of $\sigma_1(i)$
A transverse example: perfect sorting by reversals

Idea of the algorithm on an example

$\sigma_1 = 5 \ 6 \ 7 \ 9 \ 4 \ 3 \ 1 \ 2 \ 8 \ 10 \ 17 \ 13 \ 15 \ 12 \ 11 \ 14 \ 18 \ 19 \ 16$

Linear: copy the sign

Mathilde Bouvel

Some algorithmic and combinatorial problems on permutation classes
A transverse example: perfect sorting by reversals

Idea of the algorithm on an example

\[ \sigma_1 = 5 \ 6 \ 7 \ 9 \ 4 \ 3 \ 1 \ 2 \ 8 \ 10 \ 17 \ 13 \ 15 \ 12 \ 11 \ 14 \ 18 \ 19 \ 16 \]

Prime with linear parent:

Mathilde Bouvel

Some algorithmic and combinatorial problems on permutation classes
A transverse example: perfect sorting by reversals

Idea of the algorithm on an example

\[ \sigma_1 = 5 \ 6 \ 7 \ 9 \ 4 \ 3 \ 1 \ 2 \ 8 \ 10 \ 17 \ 13 \ 15 \ 12 \ 11 \ 14 \ 18 \ 19 \ 16 \]

Prime with prime parent: ???
A transverse example: perfect sorting by reversals

Idea of the algorithm on an example

$$\sigma_1 = 5 \; 6 \; 7 \; 9 \; 4 \; 3 \; 1 \; 2 \; 8 \; 10 \; 17 \; 13 \; 15 \; 12 \; 11 \; 14 \; 18 \; 19 \; 16$$
Complexity results

Previous results [BBCP07] :
- $O(2^p n \sqrt{n \log n})$, where $p =$ number of prime nodes
- polynomial on separable permutations ($p = 0$)

Complexity analysis [B., Chauve, Mishna & Rossin 09] :
- polynomial with probability 1 asymptotically
- polynomial on average
- in a parsimonious scenario for separable permutations
  - average number of reversals $\sim 1.2n$
  - average size of a reversal $\sim 1.02 \sqrt{n}$

Probability distribution : always uniforme
“Average” shape of decomposition trees

Enumeration of simple permutations: asymptotically $\frac{n!}{e^2}$

$\Rightarrow$ Asymptotically, a proportion $\frac{1}{e^2}$ of decomposition trees are reduced to one prime node.

Thm: Asymptotically, the proportion of decomposition trees made of a prime root with children that are leaves or twins is 1

twin = linear node with only two children, that are leaves

Consequence: Asymptotically, with probability 1, the algorithm runs in polynomial time.
A transverse example: perfect sorting by reversals

Average complexity

Average complexity on permutations of size $n$:

$$\sum_{p=0}^{n} \# \{ \sigma \text{ with } p \text{ prime nodes} \} \cdot C \cdot 2^p n \sqrt{n \log n}$$

$$\frac{n!}{n!}$$

Thm: When $p \geq 2$,
number of permutations of size $n$ with $p$ prime nodes $\leq \frac{48(n-1)!}{2^p}$

Consequence: Average complexity on permutations of size $n$ is
$\leq 50Cn\sqrt{n \log n}$.
In particular, polynomial on average.
A transverse example: perfect sorting by reversals

Parameters for separable permutations

Schröder trees $\approx$ decomposition trees of separable permutations:
- Average number of internal nodes: $\sim \frac{n}{\sqrt{2}}$
- Average value of the sum of the sizes of all subtrees:
  $\sim 2^{3/4} \sqrt{3 - 2\sqrt{2}} \sqrt{\pi} n^3$

Signed separable permutations:
- Average number of reversals: $\sim \frac{1+\sqrt{2}}{2} n$
- Average value of the sum of the sizes of all reversals:
  $\sim 2^{3/4} \sqrt{3 - 2\sqrt{2}} \sqrt{\pi} n^3$
- Average size of a reversal: $\sim \frac{2^{7/4} \sqrt{3 - 2\sqrt{2}}}{1+\sqrt{2}} \sqrt{\pi} n \sim 1.02 \sqrt{n}$
# Outline

1. Objects studied: Permutations, Patterns and Classes
2. Main tool: decomposition trees
3. Applications in algorithmics
4. Structure of permutations classes in combinatorics
5. A transverse example: perfect sorting by reversals
6. Conclusion and perspectives
Conclusions

With decomposition trees:

- Parametrized algorithms for finding patterns
  - pattern matching
  - longest common pattern [BR06, BRV07]
- Combinatorial study of pin-permutations
  - example of a permutation class [BBR09]
  - application for detecting structure [BBPR]
- Complexity analysis of algorithms
  - perfect sorting by reversals [BCMR09]

But also:

- Limits in the problem of finding longest common patterns, with patterns restricted to a class [B., Rossin & Vialette 07]
- Combinatorial study of the model of tandem duplication - random loss [B. et Rossin 09] [B. & Pergola 08]
Pattern matching: \textit{NP}-hard. Does there exist an algorithm \textit{polynomial in} $n$ with a preprocessing of the pattern?

- Computation of \textit{generating functions} of $S(B)$ when containing a finite number of simples: some steps still missing.

- Application to \textit{random generation}.

- Precise \textit{analysis} of other algorithms involving decomposition trees (\textit{Double-Cut and Join}).

- Extend concepts and results from \textit{graph theory} to permutations, and vice-versa.
Perspectives

- Pattern matching: $NP$-hard. Does there exist an algorithm polynomial in $n$ with a preprocessing of the pattern?
- Computation of generating functions of $S(B)$ when containing a finite number of simples: some steps still missing
- Application to random generation
- Precise analysis of other algorithms involving decomposition trees (*Double-Cut and Join*)
- Extend concepts and results from graph theory to permutations, and vice-versa

Thank you!
Pattern matching: $NP$-hard. Does there exist an algorithm polynomial in $n$ with a preprocessing of the pattern?

Computation of generating functions of $S(B)$ when containing a finite number of simples: some steps still missing

Application to random generation

Precise analysis of other algorithms involving decomposition trees (Double-Cut and Join)

Extend concepts and results from graph theory to permutations, and vice-versa

Thank you!