Factorising Pattern-Free Permutations

Édouard Bonnet Romain Bourneuf
Colin Geniet Stéphan Thomassé

ENS Lyon

LoGAlg 2023

Teaser for LoGAlg

Theorem (Folklore)

A class $\mathcal{C}$ has bounded cliquewidth iff $\mathcal{C}$ is CMSO transduction of trees.

Teaser for LoGAlg

Theorem (Folklore)

A class $\mathcal{C}$ has bounded cliquewidth iff $\mathcal{C}$ is CMSO transduction of graphs of cliquewidth 1.

Teaser for LoGAlg

Theorem (Folklore)

A class $\mathcal{C}$ has bounded cliquewidth iff $\mathcal{C}$ is CMSO transduction of graphs of cliquewidth 1.

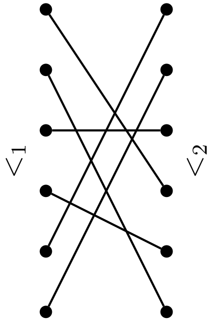
We prove:

Theorem

A class $\mathcal{C}$ has bounded twin-width iff $\mathcal{C}$ is FO transduction of graphs of twin-width c, for some universal constant c.

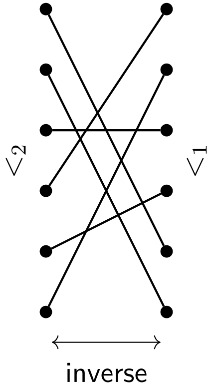
And Now for Something Completely Different

Permutation = two linear orders on the same set: $(X, <_1, <_2)$



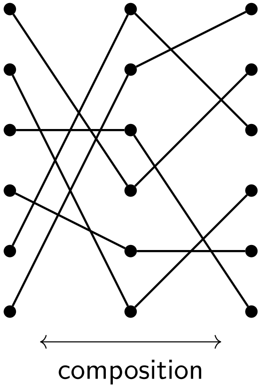
And Now for Something Completely Different

Permutation = two linear orders on the same set: $(X, <_1, <_2)$



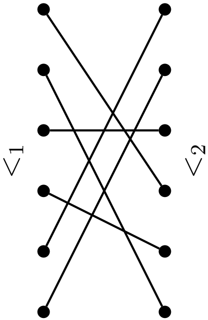
And Now for Something Completely Different

Permutation = two linear orders on the same set: $(X, <_1, <_2)$



And Now for Something Completely Different

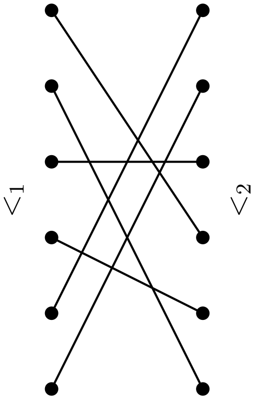
Permutation = two linear orders on the same set: $(X, <_1, <_2)$



Theorem (BNdMST '21)

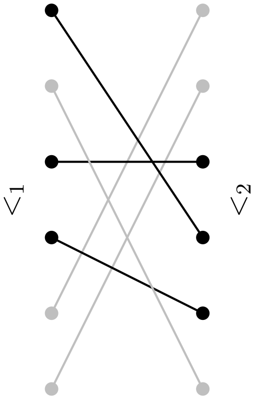
For any class $\mathcal{C}$ of bounded twin-width, there exists $\mathcal{C}'$ class of permutations of bounded twin-width such that $\mathcal{C}'$ transduces $\mathcal{C}$.

Patterns in permutations



Permutation $(X, <_1, <_2)$

Patterns in permutations

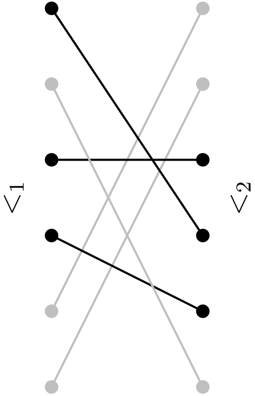


Permutation $(X, <_1, <_2)$

Pattern $=$ induced substructure

$$(Y, <_1, <_2), \quad Y \subset X$$

Patterns in permutations



Permutation $(X, <_1, <_2)$

Pattern = induced substructure

$$(Y, <_1, <_2), \quad Y \subset X$$

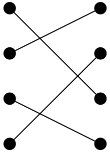
Pattern-free permutation class:

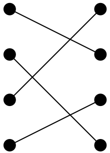
$$\mathcal{F}(\tau) = \{\sigma \ : \ \tau \not\subseteq \sigma\}$$

Separable permutations

Separable permutations $= \mathcal{F}(3142, 2413)$

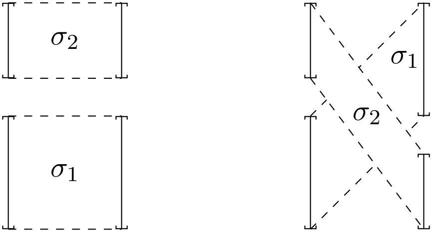
Forbidden patterns:



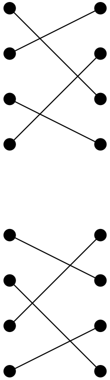


Separable permutations

Separable permutations $= \mathcal{F}(3142, 2413)$
$=$ permutations created by direct/skew sum



Forbidden patterns:



Pattern-free classes are nice

Theorem (Marcus–Tardos '04)

For any τ, there is a constant c such that $\mathcal{F}(\tau)$ has $\leq c^n$ permutations of size n.

Pattern-free classes are nice

Theorem (Marcus–Tardos '04)

For any τ, there is a constant c such that $\mathcal{F}(\tau)$ has $\leq c^n$ permutations of size n.

Recognition algorithm:

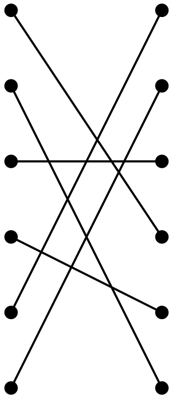
Theorem (Guillemot–Marx '14)

One can test if τ is a pattern of σ in time $f(\tau) \cdot |\sigma|$.

Twin-width

Twin-width of $(X, <_1, <_2)$:

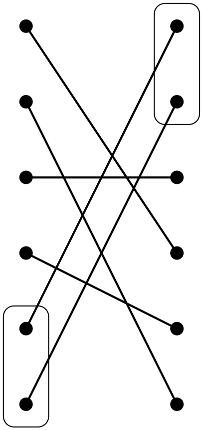
- iteratively merge elements of X
- error between $A, B \subset X$ if they interleave for either $<_1$ or $<_2$
- minimize the error degree



Twin-width

Twin-width of $(X, <_1, <_2)$:

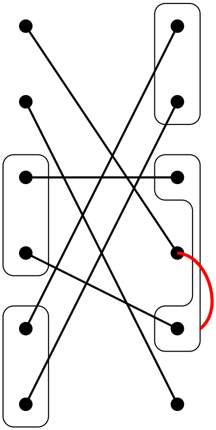
- iteratively merge elements of X
- error between $A, B \subset X$ if they interleave for either $<_1$ or $<_2$
- minimize the error degree



Twin-width

Twin-width of $(X, <_1, <_2)$:

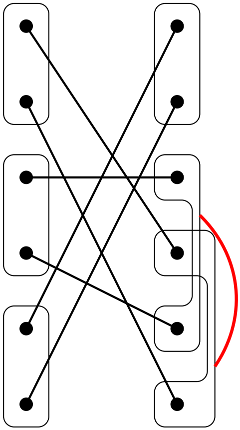
- iteratively merge elements of X
- error between $A, B \subset X$ if they interleave for either $<_1$ or $<_2$
- minimize the error degree



Twin-width

Twin-width of $(X, <_1, <_2)$:

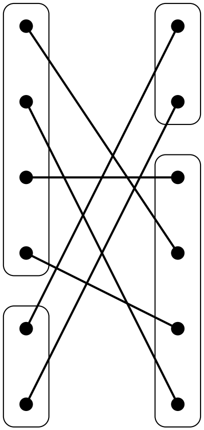
- iteratively merge elements of X
- error between $A, B \subset X$ if they interleave for either $<_1$ or $<_2$
- minimize the error degree



Twin-width

Twin-width of $(X, <_1, <_2)$:

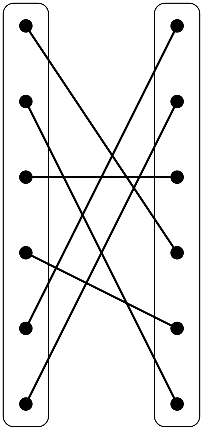
- iteratively merge elements of X
- error between $A, B \subset X$ if they interleave for either $<_1$ or $<_2$
- minimize the error degree



Twin-width

Twin-width of $(X, <_1, <_2)$:

- iteratively merge elements of X
- error between $A, B \subset X$ if they interleave for either $<_1$ or $<_2$
- minimize the error degree



Guillemot–Marx Algorithm

Theorem (Guillemot–Marx '14)

One can test if τ *is a pattern of* σ *in time* $f(\tau) \cdot |\sigma|$.

Win–win argument:

Lemma

A class $\mathcal{C}$ *avoids a pattern if and only if it has bounded twin-width.*

Lemma

One can test if τ *is a pattern of* σ *in time* $f(\tau, \mathrm{tww}(\sigma)) \cdot |\sigma|$.

Pattern-free classes are nice

Theorem (Marcus–Tardos '04)

For any τ, there is a constant c such that $\mathcal{F}(\tau)$ has $\leq c^n$ permutations of size n.

Recognition algorithm:

Theorem (Guillemot–Marx '14)

One can test if τ is a pattern of σ in time $f(\tau) \cdot |\sigma|$.

We give a 'decomposition':

Theorem (BBGT)

For any pattern τ, there is a constant k such that any $\sigma \in \mathcal{F}(\tau)$ factorises as $\sigma = \sigma_1 \circ \cdots \circ \sigma_k$, with σ_i separable.

Pattern-free classes are nice

Theorem (Marcus–Tardos '04)

For any τ, there is a constant c such that $\mathcal{F}(\tau)$ has $\leq c^n$ permutations of size n.

Recognition algorithm:

Theorem (Guillemot–Marx '14)

One can test if τ is a pattern of σ in time $f(\tau) \cdot |\sigma|$.

We give a 'decomposition':

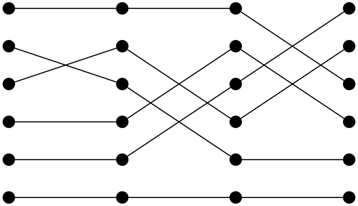
Theorem (BBGT)

For any $t \in \mathbb{N}$, there is a constant k such that any σ with $\mathrm{tww}(\sigma) \leq t$ factorises as $\sigma = \sigma_1 \circ \cdots \circ \sigma_k$, with $\mathrm{tww}(\sigma_i) = 0$.

Factorisation

Theorem

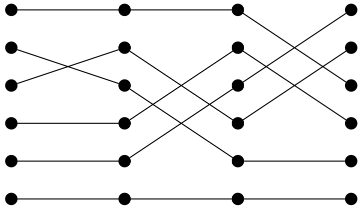
For any $t \in \mathbb{N}$, there is a constant k such that any σ with $\operatorname{tww}(\sigma) \leq t$ factorises as $\sigma = \sigma_1 \circ \cdots \circ \sigma_k$, with $\operatorname{tww}(\sigma_i) = 0$.



Factorisation

Theorem

For any $t \in \mathbb{N}$, there is a constant k such that any σ with $\mathrm{tww}(\sigma) \leq t$ factorises as $\sigma = \sigma_1 \circ \cdots \circ \sigma_k$, with $\mathrm{tww}(\sigma_i) = 0$.

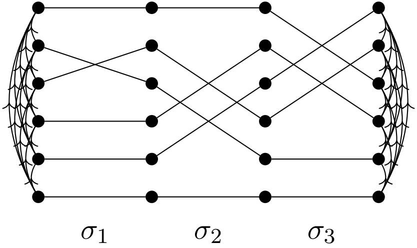


Fact

For any σ_1, σ_2, $\mathrm{tww}(\sigma_1 \circ \sigma_2) \leq f(\mathrm{tww}(\sigma_1), \mathrm{tww}(\sigma_2))$.

Transducing classes of bounded twin-width

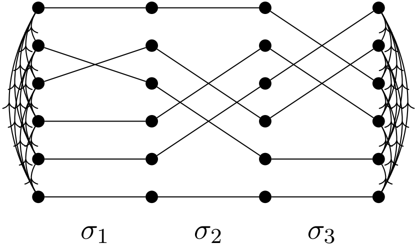
Path representation of the factorisation:



- If $\sigma_1, \ldots, \sigma_k$ are separable, the path representation has twin-width $O(1)$ (independent of k).

Transducing classes of bounded twin-width

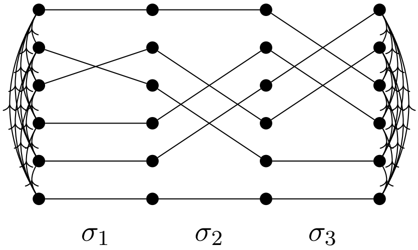
Path representation of the factorisation:



- If $\sigma_1, \ldots, \sigma_k$ are separable, the path representation has twin-width $O(1)$ (independent of k).
- For any k, there is a FO transduction reconstructing $\sigma_1 \circ \cdots \circ \sigma_k$ from the path decomposition.

Transducing classes of bounded twin-width

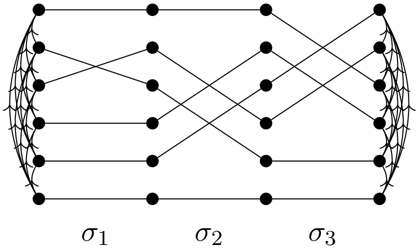
Path representation of the factorisation:



- If $\sigma_1, \ldots, \sigma_k$ are separable, the path representation has twin-width $O(1)$ (independent of k).
- For any k, there is a FO transduction reconstructing $\sigma_1 \circ \cdots \circ \sigma_k$ from the path decomposition.

Transducing classes of bounded twin-width

Path representation of the factorisation:



- If $\sigma_1, \ldots, \sigma_k$ are separable, the path representation has twin-width $O(1)$ (independent of k).
- For any k, there is a FO transduction reconstructing $\sigma_1 \circ \cdots \circ \sigma_k$ from the path decomposition.

path representation $\longrightarrow$ permutation $\longrightarrow$ graph

Proof overview (from very far away)

Theorem

For any $t \in \mathbb{N}$, there is a constant k such that any σ with $\mathrm{tww}(\sigma) \leq t$ factorises as $\sigma = \sigma_1 \circ \cdots \circ \sigma_k$, with $\mathrm{tww}(\sigma_i) = 0$.

Main tool for the proof:

Theorem (Pilipczuk, Sokołowski, Bourneuf, Thomassé, '23)

Classes of graphs with bounded twin-width are polynomially χ-bounded.

Proof overview (from very far away)

Theorem

For any $t \in \mathbb{N}$, there is a constant k such that any σ with $\mathrm{tww}(\sigma) \leq t$ factorises as $\sigma = \sigma_1 \circ \cdots \circ \sigma_k$, with $\mathrm{tww}(\sigma_i) = 0$.

Main tool for the proof:

Theorem (Pilipczuk, Sokołowski, Bourneuf, Thomassé, '23)

Classes of graphs with bounded twin-width are polynomially χ-bounded.

Behind this theorem: decomposition of graphs of twin-width k into graphs of twin-width $k - 1$.

Proof overview (from very far away)

Theorem

For any $t \in \mathbb{N}$, there is a constant k such that any σ with $\mathrm{tww}(\sigma) \leq t$ factorises as $\sigma = \sigma_1 \circ \cdots \circ \sigma_k$, with $\mathrm{tww}(\sigma_i) = 0$.

Main tool for the proof:

Theorem (Pilipczuk, Sokołowski, Bourneuf, Thomassé, '23)

Classes of graphs with bounded twin-width are polynomially χ-bounded.

Behind this theorem: decomposition of ~~graphs~~ things of twin-width k into things of twin-width $k-1$.

Proof overview (from very far away)

Theorem

For any $t \in \mathbb{N}$, there is a constant k such that any σ with $\mathrm{tww}(\sigma) \leq t$ factorises as $\sigma = \sigma_1 \circ \cdots \circ \sigma_k$, with $\mathrm{tww}(\sigma_i) = 0$.

Main tool for the proof:

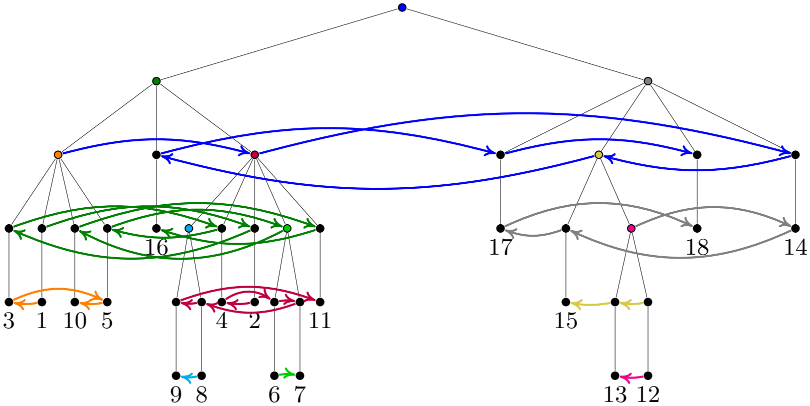
Theorem (Pilipczuk, Sokołowski, Bourneuf, Thomassé, '23)

Classes of graphs with bounded twin-width are polynomially χ-bounded.

Behind this theorem: decomposition of ~~graphs~~ things of twin-width k into things of twin-width $k-1$.

For permutations, this decomposition can be expressed with direct and skew sums, and a bounded number of products.

The Decomposition



Open Questions

- What is the smallest c such that $\mathrm{tww} = c$ transduces all of bounded twin-width (conjecture: $c = 4$).
- Applications of this factorisation?
- Computing shortest factorisations into separable permutations? (is it FPT? approximation?)
- Generalisation to matrices?