Twin-width of ordered structures

Édouard Bonnet, Ugo Giocanti, Patrice Ossona de Mendez, Pierre Simon, Stéphan Thomassé, and Szymon Toruńczyk

Université Grenoble Alpes, Laboratoire G-SCOP

Umeå February 2023

Overview

[BKTW 20] Twin-width I: tractable FO model checking

[BGKTW 20] Twin-width II: small classes

[BGKTW 20] Twin-width III: max independent set, min dominating set, and coloring

[BGOSTT 21] Twin-width IV: ordered graphs and matrices

[BGOT 22] Twin-width V: linear minors, modular counting, and matrix multiplication

[BKRT 21] Twin-width VI: the lens of contraction sequences

[BGTT 22] Twin-width VII: groups

[BCKKLT 22] Twin-width VIII: delineation and win-wins

http://perso.ens-lyon.fr/edouard.bonnet/twinwidth.html

Overview

[BKTW 20] Twin-width I: tractable FO model checking

[BGKTW 20] Twin-width II: small classes

[BGKTW 20] Twin-width III: max independent set, min dominating set, and coloring

[BGOSTT 21] Twin-width IV: ordered graphs and matrices

[BGOT 22] Twin-width V: linear minors, modular counting, and matrix multiplication

[BKRT 21] Twin-width VI: the lens of contraction sequences

[BGTT 22] Twin-width VII: groups

[BCKKLT 22] Twin-width VIII: delineation and win-wins

`http://perso.ens-lyon.fr/edouard.bonnet/twinwidth.html`

Definition (Contraction sequence)

Contraction sequence of $G = (V, E)$: sequence of trigraphs $(G = G_n, G_{n-1}, \dots, G_1)$ where G_{i-1} is obtained by identifying two vertices of G_i.

Twin-width of unordered graphs

Definition (Contraction sequence)

Contraction sequence of $G = (V, E)$: sequence of trigraphs $(G = G_n, G_{n-1}, \dots, G_1)$ where G_{i-1} is obtained by identifying two vertices of G_i.
$V(G_i) \leftrightarrow$ partition of $V(G)$.

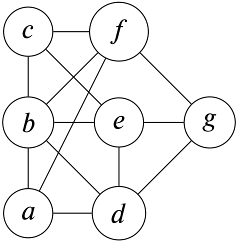
Twin-width of unordered graphs

Definition (Contraction sequence)

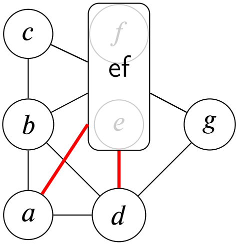
Contraction sequence of $G = (V, E)$: sequence of trigraphs $(G = G_n, G_{n-1}, \dots, G_1)$ where G_{i-1} is obtained by identifying two vertices of G_i.
$V(G_i) \leftrightarrow$ partition of $V(G)$.
For every $X, Y \in V(G_i)$ put:

- An edge $XY \in E(G_i)$ if $G[X, Y]$ is a biclique;
- A nonedge in G_i if $G[X, Y]$ has no edge;
- A red edge $XY \in R(G_i)$ otherwise.

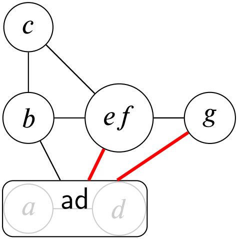
Twin-width of unordered graphs



A contraction sequence of G:
Sequence of trigraphs $G = G_n, G_{n-1}, \ldots, G_2, G_1$ such that G_i is obtained by performing one contraction in G_{i+1}.

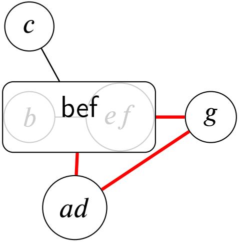


A contraction sequence of G:
Sequence of trigraphs $G = G_n, G_{n-1}, \ldots, G_2, G_1$ such that G_i is obtained by performing one contraction in G_{i+1}.



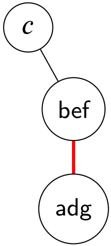
A contraction sequence of G:
Sequence of trigraphs $G = G_n, G_{n-1}, \dots, G_2, G_1$ such that G_i is obtained by performing one contraction in G_{i+1}.

Twin-width of unordered graphs



A contraction sequence of G:
Sequence of trigraphs $G = G_n, G_{n-1}, \ldots, G_2, G_1$ such that
G_i is obtained by performing one contraction in G_{i+1}.

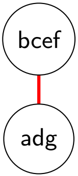
Twin-width of unordered graphs



A contraction sequence of G:

Sequence of trigraphs $G = G_n, G_{n-1}, \ldots, G_2, G_1$ such that G_i is obtained by performing one contraction in G_{i+1}.

Twin-width of unordered graphs



A contraction sequence of G:
Sequence of trigraphs $G = G_n, G_{n-1}, \dots, G_2, G_1$ such that G_i is obtained by performing one contraction in G_{i+1}.

A contraction sequence of G:
Sequence of trigraphs $G = G_n, G_{n-1}, \dots, G_2, G_1$ such that G_i is obtained by performing one contraction in G_{i+1}.

Twin-width of unordered graphs

Definition (Contraction sequence,twin-width)

Contraction sequence of $G = (V, E)$: sequence of trigraphs $(G = G_n, G_{n-1}, \dots, G_1)$ where G_{i-1} is obtained by identifying two vertices of G_i.
$V(G_i) \leftrightarrow$ partition of $V(G)$.
For every $X, Y \in V(G_i)$ put:

- An edge if $G[X, Y]$ is a biclique;
- A nonedge if $G[X, Y]$ has no edge;
- A red edge otherwise.

$(G_i)_i$ has width at most d if every G_i has red degree at most d.
The twin-width of G is the minimum width a contraction sequence of G could have.

Warnings

- "Vertex-contraction" means "vertex-identification".

Warnings

- "Vertex-contraction" means "vertex-identification".
- Twin-width is not decreasing when taking subgraphs.

Warnings

- "Vertex-contraction" means "vertex-identification".
- Twin-width is not decreasing when taking subgraphs. However it changes when considering induced subgraphs: for every $H \leq_{ind} G$, $tww(H) \leq tww(G)$.

Examples and properties

- Cographs $\Leftrightarrow$ Graphs with twin-width 0;

Examples and properties

- Cographs ⇔ Graphs with twin-width 0;
- Trees have twin-width at most 2;

Examples and properties

- Cographs ⇔ Graphs with twin-width 0;
- Trees have twin-width at most 2;
- [JP 22] Graphs of treewidth t have twin-width at most $3 \cdot 2^{t-1}$;

Examples and properties

- Cographs ⇔ Graphs with twin-width 0;
- Trees have twin-width at most 2;
- [JP 22] Graphs of treewidth t have twin-width at most $3 \cdot 2^{t-1}$;
- [HJ 22] Planar graphs have twin-width at most 8;

Examples and properties

- Cographs $\Leftrightarrow$ Graphs with twin-width 0;
- Trees have twin-width at most 2;
- [JP 22] Graphs of treewidth t have twin-width at most $3 \cdot 2^{t-1}$;
- [HJ 22] Planar graphs have twin-width at most 8;
- K_t-minor free graphs have twin-width $2^{2^{2^{\mathcal{O}(t)}}}$;

Examples and properties

- Cographs $\Leftrightarrow$ Graphs with twin-width 0;
- Trees have twin-width at most 2;
- [JP 22] Graphs of treewidth t have twin-width at most $3 \cdot 2^{t-1}$;
- [HJ 22] Planar graphs have twin-width at most 8;
- K_t-minor free graphs have twin-width $2^{2^{2^{\mathcal{O}(t)}}}$;
- Graphs with clique-width t have twin-width $2^{\mathcal{O}(t)}$;

Examples and properties

- Cographs $\Leftrightarrow$ Graphs with twin-width 0;
- Trees have twin-width at most 2;
- [JP 22] Graphs of treewidth t have twin-width at most $3 \cdot 2^{t-1}$;
- [HJ 22] Planar graphs have twin-width at most 8;
- K_t-minor free graphs have twin-width $2^{2^{2^{\mathcal{O}(t)}}}$;
- Graphs with clique-width t have twin-width $2^{\mathcal{O}(t)}$;
- Permutation graphs G_σ such that σ avoids a pattern τ have twin-width $2^{\mathcal{O}(|\tau|)}$;

Examples and properties

- Cographs $\Leftrightarrow$ Graphs with twin-width 0;
- Trees have twin-width at most 2;
- [JP 22] Graphs of treewidth t have twin-width at most $3 \cdot 2^{t-1}$;
- [HJ 22] Planar graphs have twin-width at most 8;
- K_t-minor free graphs have twin-width $2^{2^{2^{\mathcal{O}(t)}}}$;
- Graphs with clique-width t have twin-width $2^{\mathcal{O}(t)}$;
- Permutation graphs G_σ such that σ avoids a pattern τ have twin-width $2^{\mathcal{O}(|\tau|)}$;
- …

Theorem (Bergé, Bonnet, Déprés 22)

Deciding whether a given graph has twin-width at most 4 *is NP-Complete.*

Algorithmic aspect of twin-width

Theorem (Bergé, Bonnet, Déprés 22)

Deciding whether a given graph has twin-width at most 4 is NP-Complete.

Question (Fundamental question of twin-width)

Can we approximate twin-width?

Algorithmic aspect of twin-width

Theorem (Bergé, Bonnet, Déprés 22)

Deciding whether a given graph has twin-width at most 4 *is NP-Complete.*

Question (Fundamental question of twin-width)

Can we approximate twin-width? i.e. Is there an algorithm taking d, G *as input and returning in time* $f(d) \cdot n^{\mathcal{O}(1)}$ *either a "No" answer if* G *has twin-width more than* d*, or an* $f(d)$*-sequence otherwise?*

Theorem (Bergé, Bonnet, Déprés 22)

Deciding whether a given graph has twin-width at most 4 is NP-Complete.

Question (Fundamental question of twin-width)

Can we approximate twin-width? i.e. Is there an algorithm taking d, G as input and returning in time $f(d) \cdot n^{\mathcal{O}(1)}$ either a "No" answer if G has twin-width more than d, or an $f(d)$-sequence otherwise?

Positive answer for every known "interesting family" of bounded twin-width.

$\varphi \in \mathrm{FO}(E^{(2)})$: first order formula describing a graph problem.

$\varphi \in \mathrm{FO}(E^{(2)})$: first order formula describing a graph problem.

Example

$$\varphi := \exists x_1, \exists x_2, \ldots, \exists x_k, \forall x, \left(\bigvee_{i=1}^{k} x = x_i\right) \vee \left(\bigvee_{i=1}^{k} E(x, x_i)\right)$$

corresponds to k-Dominating Set problem.

FO model checking on graphs

$\varphi \in \mathrm{FO}(E^{(2)})$: first order formula describing a graph problem.

Example

H: fixed graph with $V(H) = \{v_1, \dots, v_k\}$.

$$\varphi_H := \exists x_1, \exists x_2, \dots, \exists x_k,$$
$$\left(\bigwedge_{1\le i<j\le k} x_i \neq x_j\right) \wedge \left(\bigwedge_{v_iv_j\in E(H)} E(x_i,x_j)\right) \wedge \left(\bigwedge_{v_iv_j\in E\left(\overline{H}\right)} \neg E(x_i,x_j)\right)$$

FO model checking on graphs

$\varphi \in \mathrm{FO}(E^{(2)})$: first order formula describing a graph problem.

Example

H: fixed graph with $V(H) = \{v_1, \dots, v_k\}$.

$$\varphi_H := \exists x_1, \exists x_2, \dots, \exists x_k,$$

$$\left(\bigwedge_{1\le i<j\le k} x_i \neq x_j\right) \wedge \left(\bigwedge_{v_iv_j \in E(H)} E(x_i, x_j)\right) \wedge \left(\bigwedge_{v_iv_j \in E\left(\overline{H}\right)} \neg E(x_i, x_j)\right)$$

corresponds to the H-Induced Subgraph problem.

$\varphi \in \mathrm{FO}(E^{(2)})$: first order formula describing a graph problem.

Definition

A class of graphs $\mathcal{C}$ is FO-FPT if there is an algorithm deciding for every $G \in \mathcal{C}$ whether $G \vDash \varphi$ in time $\mathcal{O}(f(|\varphi|) \cdot n^{\mathcal{O}(1)})$ for some computable f.

FO model checking on graphs

$\varphi \in \mathrm{FO}(E^{(2)})$: first order formula describing a graph problem.

Definition

A class of graphs $\mathcal{C}$ is FO-FPT if there is an algorithm deciding for every $G \in \mathcal{C}$ whether $G \models \varphi$ in time $\mathcal{O}(f(|\varphi|) \cdot n^{\mathcal{O}(1)})$ for some computable f.

Theorem (Bonnet, Kim, Thomassé, Watrigant '20)

There exists an algorithm that, given a graph G, a certificate that $tww(G) \leq d$ and a formula φ, decides whether $G \models \varphi$ in time $\mathcal{O}(f(d, |\varphi|) \cdot n)$.

FO model checking on graphs

$\varphi \in \mathrm{FO}(E^{(2)})$: first order formula describing a graph problem.

Definition

A class of graphs $\mathcal{C}$ is FO-FPT if there is an algorithm deciding for every $G \in \mathcal{C}$ whether $G \vDash \varphi$ in time $\mathcal{O}(f(|\varphi|) \cdot n^{\mathcal{O}(1)})$ for some computable f.

Theorem (Bonnet, Kim, Thomassé, Watrigant '20)

There exists an algorithm that, given a graph G, a certificate that $tww(G) \leq d$ and a formula φ, decides whether $G \vDash \varphi$ in time $\mathcal{O}(f(d, |\varphi|) \cdot n)$.

Remark

Existence of an approximation algorithm for twin-width $\Rightarrow$ classes of bounded twin-width are FO-FPT.

FO+MOD model checking on graphs

$\varphi \in \mathrm{FO} + \mathrm{MOD}(E^{(2)})$: first order formula describing a graph problem where we also allow existential quantifiers $\exists^{i[p]} x, \phi(x)$ expressing "there exists i mod p witnesses x for ϕ".

FO+MOD model checking on graphs

$\varphi \in \mathrm{FO} + \mathrm{MOD}(E^{(2)})$: first order formula describing a graph problem where we also allow existential quantifiers $\exists^{i[p]} x, \phi(x)$ expressing "there exists i mod p witnesses x for ϕ".

Example

$$\varphi^2(x, y) := \exists^{1[2]} z, E(x, z) \wedge E(z, y)$$

"there exists an odd number of xy-paths of size 2".

FO+MOD model checking on graphs

$\varphi \in \mathrm{FO} + \mathrm{MOD}(E^{(2)})$: first order formula describing a graph problem where we also allow existential quantifiers $\exists^{i[p]} x, \phi(x)$ expressing "there exists i mod p witnesses x for ϕ".

Example

$$\varphi^2(x, y) := \exists^{1[2]} z, E(x, z) \land E(z, y)$$

"there exists an odd number of xy-paths of size 2".

Example

$$\psi(x, y) := \exists^{0[2]} z, E(x, z) \land \varphi^2(z, y)$$

FO+MOD model checking on graphs

$\varphi \in \mathrm{FO} + \mathrm{MOD}(E^{(2)})$: first order formula describing a graph problem where we also allow existential quantifiers $\exists^{i[p]} x, \phi(x)$ expressing "there exists i mod p witnesses x for ϕ".

Example

$$\varphi^2(x, y) := \exists^{1[2]} z, E(x, z) \wedge E(z, y)$$

"there exists an odd number of xy-paths of size 2".

Example

$$\psi(x, y) := \exists^{0[2]} z, E(x, z) \wedge \varphi^2(z, y)$$

"there exists an even number of xy-paths of size 3".

FO+MOD model checking on graphs

$\varphi \in \mathrm{FO} + \mathrm{MOD}(E^{(2)})$: first order formula describing a graph problem where we also allow existential quantifiers $\exists^{i[p]} x, \phi(x)$ expressing "there exists i mod p witnesses x for ϕ".

Definition

A class of graphs $\mathcal{C}$ is (FO+MOD)-FPT if there is an algorithm deciding for every $G \in \mathcal{C}$ whether $G \models \varphi$ in time $\mathcal{O}(f(|\varphi|) \cdot n^{\mathcal{O}(1)})$ for some computable f.

FO+MOD model checking on graphs

$\varphi \in \mathrm{FO}+\mathrm{MOD}(E^{(2)})$: first order formula describing a graph problem where we also allow existential quantifiers $\exists^{i[p]}x, \phi(x)$ expressing "there exists i mod p witnesses x for ϕ".

Definition

A class of graphs $\mathcal{C}$ is (FO+MOD)-FPT if there is an algorithm deciding for every $G \in \mathcal{C}$ whether $G \vDash \varphi$ in time $\mathcal{O}(f(|\varphi|) \cdot n^{\mathcal{O}(1)})$ for some computable f.

Theorem (BKTW 20, BGOT 22)

There exists an algorithm that, given a graph G, a certificate that $tww(G) \leq d$ and a FO+MOD formula φ, decides whether $G \vDash \varphi$ in time $\mathcal{O}(f(d, |\varphi|) \cdot n)$.

Interpretations and transductions

Interpretation: $\varphi(x, y) \in FO(E^{(2)})$ (or $FO + MOD(E^{(2)})$) on two free variables x, y. For every graph G, define $\varphi(G)$ on vertex set $V(G)$ and edge set: $E(\varphi(G)) := \{uv, G \models \varphi(u, v)\}$.

Interpretation: $\varphi(x, y) \in FO(E^{(2)})$ (or $FO + MOD(E^{(2)})$) on two free variables x, y. For every graph G, define $\varphi(G)$ on vertex set $V(G)$ and edge set: $E(\varphi(G)) := \{uv, G \models \varphi(u, v)\}$.

Example

$$\varphi(x, y) = \neg E(x, y)$$

$\varphi(G) = \overline{G}$: complement graph.

Interpretations and transductions

Interpretation: $\varphi(x,y) \in FO(E^{(2)})$ (or $FO + MOD(E^{(2)})$) on two free variables x, y. For every graph G, define $\varphi(G)$ on vertex set $V(G)$ and edge set: $E(\varphi(G)) := \{uv, G \models \varphi(u,v)\}$.

Example

$$\varphi(x,y) = \neg E(x,y)$$

$\varphi(G) = \overline{G}$: complement graph.

Example

$$\varphi(x,y) = E(x,y) \vee (\exists z, E(x,z) \wedge E(z,y))$$

Interpretation: $\varphi(x,y) \in FO(E^{(2)})$ (or $FO + MOD(E^{(2)})$) on two free variables x, y. For every graph G, define $\varphi(G)$ on vertex set $V(G)$ and edge set: $E(\varphi(G)) := \{uv, G \models \varphi(u,v)\}$.

Example

$$\varphi(x,y) = \neg E(x,y)$$

$\varphi(G) = \overline{G}$: complement graph.

Example

$$\varphi(x,y) = E(x,y) \vee (\exists z, E(x,z) \wedge E(z,y))$$

$\varphi(G) = G^2$: square graph.

Interpretation: $\varphi(x,y) \in FO(E^{(2)})$ (or $FO + MOD(E^{(2)})$) on two free variables x, y. For every graph G, define $\varphi(G)$ on vertex set $V(G)$ and edge set: $E(\varphi(G)) := \{uv, G \models \varphi(u,v)\}$.
$\mathcal{C}$: class of graphs $\to$ $\varphi(\mathcal{C}) := \mathrm{Clos}\{\varphi(G), G \in \mathcal{C}\}$.

Independence

$\mathcal{G} :=$ class of all finite graphs.

Independence

$\mathcal{G} :=$ class of all finite graphs.

Definition

A hereditary class $\mathcal{C}$ of graphs is independent if there exists an interpretation $\varphi(x, y) \in FO$ such that $\varphi(\mathcal{C}) = \mathcal{G}$.
Otherwise it is dependent.

$\mathcal{G}$:= class of all finite graphs.

Definition

A hereditary class $\mathcal{C}$ of graphs is independent if there exists an interpretation $\varphi(x, y) \in FO$ such that $\varphi(\mathcal{C}) = \mathcal{G}$.
Otherwise it is dependent.

Example

$\mathcal{C} := \mathrm{Clos}(\{K_n^{(1)}, n \in \mathcal{N}\})$ is independent.

Independence

$\mathcal{G}$:= class of all finite graphs.

Definition

A hereditary class $\mathcal{C}$ of graphs is independent if there exists an interpretation $\varphi(x, y) \in FO$ such that $\varphi(\mathcal{C}) = \mathcal{G}$.
Otherwise it is dependent.

Example

$\mathcal{C} := \mathrm{Clos}(\{K_n^{(1)}, n \in \mathcal{N}\})$ is independent.

$$\varphi(x, y) := \exists z, (\deg(z) \leq 2) \wedge E(x, z) \wedge E(z, y).$$

Independence

$\mathcal{G}$:= class of all finite graphs.

Definition

A hereditary class $\mathcal{C}$ of graphs is independent if there exists an interpretation $\varphi(x, y) \in FO$ such that $\varphi(\mathcal{C}) = \mathcal{G}$.
Otherwise it is dependent.

Example

$\mathcal{C} := \mathrm{Clos}(\{K_n^{(1)}, n \in \mathcal{N}\})$ is independent.

$$\varphi(x, y) := \exists z, \big(\forall z', E(z, z') \Rightarrow (z' = x) \vee (z' = y)\big) \wedge E(x, z) \wedge E(z, y).$$

Independence

$\mathcal{G}$:= class of all finite graphs.

Definition

A hereditary class $\mathcal{C}$ of graphs is independent if there exists an interpretation $\varphi(x, y) \in FO$ such that $\varphi(\mathcal{C}) = \mathcal{G}$.
Otherwise it is dependent.

Theorem (Bonnet, Kim, Thomassé, Watrigant '20)

If $\mathcal{C}$ is a class of graphs of twin-width at most t and $\varphi(x, y)$ an interpretation, then:

$$tww(\varphi(\mathcal{C})) \leq f(t, |\varphi|)$$

for some computable f.

Independence

$\mathcal{G}$:= class of all finite graphs.

Definition

A hereditary class $\mathcal{C}$ of graphs is independent if there exists an interpretation $\varphi(x, y) \in FO$ such that $\varphi(\mathcal{C}) = \mathcal{G}$.
Otherwise it is dependent.

Theorem (Bonnet, Kim, Thomassé, Watrigant '20)

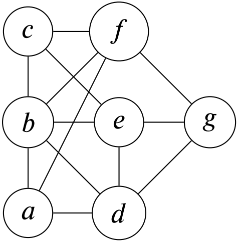
If $\mathcal{C}$ is a class of graphs of twin-width at most t and $\varphi(x, y)$ an interpretation, then:

$$tww(\varphi(\mathcal{C})) \leq f(t, |\varphi|)$$

for some computable f.

- Also true for FO+MOD interpretations;
- Classes of graphs of bounded twin-width are dependent.

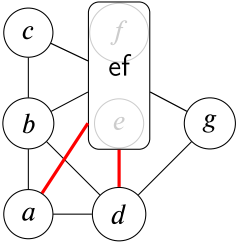
Graphs are given together with a total order on their vertices.
Equivalent to work on an adjacency matrix of G.



$$\begin{array}{c c}
\begin{matrix} g \\ f \\ e \\ d \\ c \\ b \\ a \end{matrix} & \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \\
 & \begin{matrix} a & b & c & d & e & f & g \end{matrix}
\end{array}$$

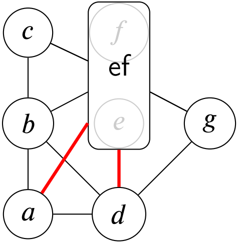
Left: Total order on $V(G)$: $a < b < c < d < e < f < g$. Right: the associated ordered adjacency matrix.

Twin-width of ordered structures



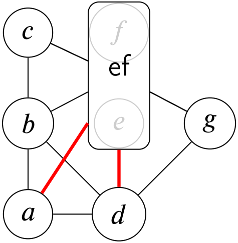
$$
\begin{array}{c}
\begin{array}{c} g \\ f \\ e \\ d \\ c \\ b \\ a \end{array}
\left[\begin{array}{ccccccc}
0 & 0 & 0 & 1 & 1 & 1 & 0 \\
1 & 1 & 1 & 0 & 0 & 0 & 1 \\
0 & 1 & 1 & 1 & 0 & 0 & 1 \\
1 & 1 & 0 & 0 & 1 & 0 & 1 \\
0 & 1 & 0 & 0 & 1 & 1 & 0 \\
1 & 0 & 1 & 1 & 1 & 1 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 0
\end{array}\right] \\
\begin{array}{ccccccc} a & b & c & d & e & f & g \end{array}
\end{array}
$$

Twin-width of ordered structures



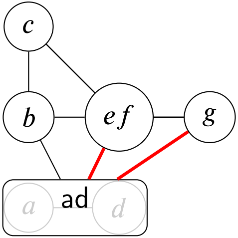
g	0	0	0	1	1	1	0
f	1	1	1	0	0	0	1
e	0	1	1	1	0	0	1
d	1	1	0	0	r	0	1
c	0	1	0	0	1	1	0
b	1	0	1	1	1	1	0
a	0	1	0	1	r	1	0
	a	b	c	d	e	f	g

Twin-width of ordered structures



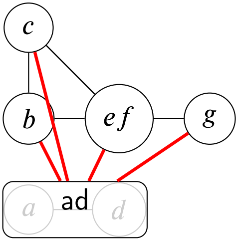
g	0	0	0	1	1	1	0
f	1	1	1	0	0	0	1
e	*r*	1	1	*r*	0	0	1
d	1	1	0	0	*r*	0	1
c	0	1	0	0	1	1	0
b	1	0	1	1	1	1	0
a	0	1	0	1	*r*	1	0
	a	b	c	d	e	f	g

Twin-width of ordered structures



	a	b	c	d	e	f	g
g	r	0	0	1	1	1	0
f	1	1	1	0	0	0	1
e	r	1	1	r	0	0	1
d	r	1	0	0	r	0	1
c	0	1	0	0	1	1	0
b	1	0	1	1	1	1	0
a	0	1	0	1	r	1	0

Twin-width of ordered structures



g	r	0	0	1	1	1	0
f	1	1	1	0	0	0	1
e	r	1	1	r	0	0	1
d	r	1	0	0	r	0	1
c	0	1	0	0	1	1	0
b	1	0	1	1	1	1	0
a	0	1	0	1	r	1	0
	a	b	c	d	e	f	g

Remark

A graph G has twin-width at most d if and only if there is a total ordering $<$ of $V(G)$ such that $(G,<)$ has twin-width at most d.

Remark

A graph G has twin-width at most d if and only if there is a total ordering $<$ of $V(G)$ such that $(G,<)$ has twin-width at most d.

Definition of twin-width can be extended to matrices with entries on a finite alphabet (e.g. $\mathbb{F}_q$).

Algorithmic aspect of twin-width for ordered structures

Theorem (BGOSTT 21)

There is an algorithm that, given an ordered graph $(G, <)$ and an integer d, returns in time $\mathcal{O}(f(d)n^2 \log(n))$:

- *"No" if $tww(G) > d$;*
- *a $g(d)$-sequence otherwise.*

Algorithmic aspect of twin-width for ordered structures

Theorem (BGOSTT 21)

There is an algorithm that, given an ordered graph $(G, <)$ and an integer d, returns in time $2^{2^{2^{2^{\mathcal{O}\left(d^2 \log(d)\right)}}}} n^2 \log(n)$:

- *"No" if $tww(G) > d$;*
- *a $2^{2^{2^{\mathcal{O}\left(d^4\right)}}}$-sequence otherwise.*

Algorithmic aspect of twin-width for ordered structures

Theorem (BGOSTT 21)

There is an algorithm that, given an ordered graph $(G, <)$ and an integer d, returns in time $\mathcal{O}(f(d)n^2 \log(n))$:

- *"No" if $tww(G) > d$;*
- *a $g(d)$-sequence otherwise.*

Characterizations of twin-width boundedness in the ordered case

Theorem (Graph version)

Let $\mathcal{C}$ be a hereditary class of ordered graphs. The following are equivalent.

1. *$\mathcal{C}$ has bounded twin-width;*
2. *$\mathcal{C}$ is FO-FPT;*
3. *$\mathcal{C}$ is (FO+MOD)-FPT;*
4. *$\mathcal{C}$ is dependent;*
5. *$\mathcal{C}$ is (FO+MOD)-dependent;*
6. *$\mathcal{C}$ contains $2^{O(n)}$ ordered n-vertex graphs.*
7. *$\mathcal{C}$ contains less than $\sum_{k=0}^{\lfloor n/2 \rfloor} \binom{n}{2k} k!$ ordered n-vertex graphs, for some n.*

Matrix multiplication

A, B: matrices over $\mathbb{F}_2$ (or $\mathbb{F}_k$).
Goal: Compute $A \cdot B$ in time $f(tww(A), tww(B)) \cdot n^{\mathcal{O}(1)}$.

Matrix multiplication

A, B: matrices over $\mathbb{F}_2$ (or $\mathbb{F}_k$).
Goal: Compute $A \cdot B$ in time $f(tww(A), tww(B)) \cdot n^{\mathcal{O}(1)}$.

$$\begin{pmatrix} 0 & A \\ B & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 & A \\ B & 0 \end{pmatrix} = \begin{pmatrix} AB & 0 \\ 0 & BA \end{pmatrix}$$

allows to reduce to the problem of squaring a matrix.

Graph interpretation

Definition

G: graph. $G^{[2]}$: modular square of G, with same vertices and:

$$E(G^{[2]}) := \{uv \ : \ |N(u) \cap N(v)| = 1 \pmod 2\}.$$

Graph interpretation

Definition

G: graph. $G^{[2]}$: modular square of G, with same vertices and:

$$E(G^{[2]}) := \{uv \ : \ |N(u) \cap N(v)| = 1 \pmod 2\}.$$

Remark

The adjacency matrices of $G^{[2]}$ are square of the ones of G.

Graph interpretation

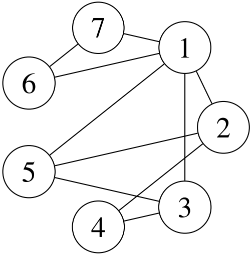
Definition

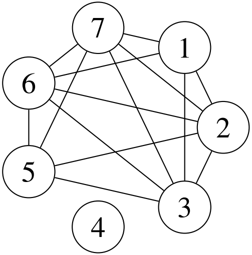
G: graph. $G^{[2]}$: modular square of G, with same vertices and:

$$E(G^{[2]}) := \{uv \ : \ |N(u) \cap N(v)| = 1 \pmod 2\}.$$

Remark

The adjacency matrices of $G^{[2]}$ are square of the ones of G.





A first "algorithm"

$\varphi^{[2]}(x, y) := \exists^{1[2]} z, E(x, z) \wedge E(z, y)$. For every G: $\varphi^{[2]}(G) = G^{[2]}$.

A first "algorithm"

$\varphi^{[2]}(x,y) := \exists^{1[2]} z, E(x,z) \wedge E(z,y)$. For every G: $\varphi^{[2]}(G) = G^{[2]}$.

Theorem

- *Given a twin-decomposition of width d of G, a twin-decomposition of width $f(d)$ of $G^{[2]}$ can be computed in time $f(d) \cdot n$.*
- *Together with approximation algorithm + previous remarks → there is a $\mathcal{O}_{d,q}(n^2 \log(n))$-time algorithm taking A, B $n \times n$ matrices and returning AB.*

A first "algorithm"

$\varphi^{[2]}(x, y) := \exists^{1[2]} z, E(x, z) \wedge E(z, y)$. For every G: $\varphi^{[2]}(G) = G^{[2]}$.

Theorem

- *Given a twin-decomposition of width d of G, a twin-decomposition of width $f(d)$ of $G^{[2]}$ can be computed in time $f(d) \cdot n$.*
- *Together with approximation algorithm + previous remarks → there is a $\mathcal{O}_{d,q}(n^2 \log(n))$-time algorithm taking A, B $n \times n$ matrices and returning AB.*

Completely unpractical

A "real" algorithm for matrix multiplication

Theorem

There exists a $\mathcal{O}(d^2 4^d n)$*-time algorithm that, given a graph* G *and a certificate that* $tww(G) \leq d$*, outputs a certificate that* $tww(G^{[2]}) = \mathcal{O}(d^2 2^d)$ *encoding* $G^{[2]}$*.*

A "real" algorithm for matrix multiplication

Theorem

There exists a $\mathcal{O}(d^2 4^d n)$-time algorithm that, given a graph G and a certificate that $tww(G) \leq d$, outputs a certificate that $tww(G^{[2]}) = \mathcal{O}(d^2 2^d)$ encoding $G^{[2]}$.

→ Extends to a FPT-algorithm for matrix multiplication over $\mathbb{F}_2$ with same complexity.

A “real” algorithm for matrix multiplication

Theorem

There exists a $\mathcal{O}(d^2 4^d n)$-time algorithm that, given a graph G and a certificate that $tww(G) \leq d$, outputs a certificate that $tww(G^{[2]}) = \mathcal{O}(d^2 2^d)$ encoding $G^{[2]}$.

→ Extends to a FPT-algorithm for matrix multiplication over $\mathbb{F}_2$ with same complexity.

→ Extends over $\mathbb{F}_q$ for q: prime power.

A "real" algorithm for matrix multiplication

Theorem

There exists a $\mathcal{O}(d^2 4^d n)$-time algorithm that, given a graph G and a certificate that $tww(G) \leq d$, outputs a certificate that $tww(G^{[2]}) = \mathcal{O}(d^2 2^d)$ encoding $G^{[2]}$.

→ Extends to a FPT-algorithm for matrix multiplication over $\mathbb{F}_2$ with same complexity.
→ Extends over $\mathbb{F}_q$ for q: prime power.
Can be combined with approximation algorithm to give a $\mathcal{O}_{d,q}(n^2 \log(n))$ algorithm computing product of matrices A and B when $tww(A), tww(B) \leq d$.

Thanks