

Parameterized Complexity is Necessary to Understand Induced Subgraph Isomorphisms

Yijia Chen
Shanghai Jiaotong University

June, 2008

Parameterized Complexity is Necessary to Understand Induced Subgraph Isomorphisms

Yijia Chen
Shanghai Jiaotong University

June, 2008

Joint Work with Marc Thurley and Mark Weyer

This talk is about

This talk is about

Find a needle in a haystack

This talk is about

Find a needle in a haystack

In Chinese: Find a needle in the bottom of a sea

This talk is about

Find a needle in a haystack

In Chinese: Find a needle in the bottom of a sea

The Moral: It is hard to find a **small object** in a **large environment**.

This talk is about

Find a needle in a haystack

In Chinese: Find a needle in the bottom of a sea

The Moral: It is hard to find a **small object** in a **large environment**.

The Question: How to prove it rigorously?

Some basic graph theory

Some basic graph theory

Let G be an **(undirected) graph**. Then $V(G)$ is its vertex set, and $E(G)$ its edge set.

Some basic graph theory

Let G be an **(undirected) graph**. Then $V(G)$ is its vertex set, and $E(G)$ its edge set.

Definition

Let G and H be two graphs. Then G is an induced subgraph of H if

Some basic graph theory

Let G be an **(undirected) graph**. Then $V(G)$ is its vertex set, and $E(G)$ its edge set.

Definition

Let G and H be two graphs. Then G is an induced subgraph of H if

- ▶ $V(G) \subseteq V(H)$;

Some basic graph theory

Let G be an **(undirected) graph**. Then $V(G)$ is its vertex set, and $E(G)$ its edge set.

Definition

Let G and H be two graphs. Then G is an induced subgraph of H if

- ▶ $V(G) \subseteq V(H)$;
- ▶ $E(G) = E(H) \upharpoonright V(G)$

Some basic graph theory

Let G be an **(undirected) graph**. Then $V(G)$ is its vertex set, and $E(G)$ its edge set.

Definition

Let G and H be two graphs. Then G is an induced subgraph of H if

- ▶ $V(G) \subseteq V(H)$;
- ▶ $E(G) = E(H) \upharpoonright V(G) = \{\{u, v\} \in E(H) \mid u, v \in V(G)\}$.

Some basic graph theory

Let G be an **(undirected) graph**. Then $V(G)$ is its vertex set, and $E(G)$ its edge set.

Definition

Let G and H be two graphs. Then G is an induced subgraph of H if

- ▶ $V(G) \subseteq V(H)$;
- ▶ $E(G) = E(H) \upharpoonright V(G) = \{\{u, v\} \in E(H) \mid u, v \in V(G)\}$.

Definition

Let G and H be two graphs. Then a mapping $f : V(G) \rightarrow V(H)$ is an isomorphism if

Some basic graph theory

Let G be an **(undirected) graph**. Then $V(G)$ is its vertex set, and $E(G)$ its edge set.

Definition

Let G and H be two graphs. Then G is an induced subgraph of H if

- ▶ $V(G) \subseteq V(H)$;
- ▶ $E(G) = E(H) \upharpoonright V(G) = \{\{u, v\} \in E(H) \mid u, v \in V(G)\}$.

Definition

Let G and H be two graphs. Then a mapping $f : V(G) \rightarrow V(H)$ is an isomorphism if

- ▶ f is a **bijection**;

Some basic graph theory

Let G be an **(undirected) graph**. Then $V(G)$ is its vertex set, and $E(G)$ its edge set.

Definition

Let G and H be two graphs. Then G is an induced subgraph of H if

- ▶ $V(G) \subseteq V(H)$;
- ▶ $E(G) = E(H) \upharpoonright V(G) = \{\{u, v\} \in E(H) \mid u, v \in V(G)\}$.

Definition

Let G and H be two graphs. Then a mapping $f : V(G) \rightarrow V(H)$ is an isomorphism if

- ▶ f is a **bijection**;
- ▶ for every $u, v \in V(G)$ we have $\{u, v\} \in E(G)$ if and only if $\{f(u), f(v)\} \in E(H)$.

Some basic graph theory

Let G be an **(undirected) graph**. Then $V(G)$ is its vertex set, and $E(G)$ its edge set.

Definition

Let G and H be two graphs. Then G is an induced subgraph of H if

- ▶ $V(G) \subseteq V(H)$;
- ▶ $E(G) = E(H) \upharpoonright V(G) = \{\{u, v\} \in E(H) \mid u, v \in V(G)\}$.

Definition

Let G and H be two graphs. Then a mapping $f : V(G) \rightarrow V(H)$ is an isomorphism if

- ▶ f is a **bijection**;
- ▶ for every $u, v \in V(G)$ we have $\{u, v\} \in E(G)$ if and only if $\{f(u), f(v)\} \in E(H)$.

Then, G and H are **isomorphic**.

Induced subgraph problems

Induced subgraph problems

Definition

Let \mathbf{C} be a class of graphs. Then the induced subgraph problem on \mathbf{C} is

STREMB(\mathbf{C})

Input: Two graphs G and H with $G \in \mathbf{C}$.

Problem: Does H contain an **induced subgraph isomorphic to G** ?

Induced subgraph problems

Definition

Let \mathbf{C} be a class of graphs. Then the induced subgraph problem on \mathbf{C} is

STREMB(\mathbf{C})

Input: Two graphs G and H with $G \in \mathbf{C}$.

Problem: Does H contain an induced subgraph isomorphic to G ?

\mathbf{C} is the class of needles, and a haystack is a graph H .

Induced subgraph problems

Definition

Let \mathbf{C} be a class of graphs. Then the induced subgraph problem on \mathbf{C} is

STREMB(\mathbf{C})

Input: Two graphs G and H with $G \in \mathbf{C}$.

Problem: Does H contain an induced subgraph isomorphic to G ?

\mathbf{C} is the class of needles, and a haystack is a graph H .

Proviso: We always assume that \mathbf{C} is recursively enumerable:

Induced subgraph problems

Definition

Let \mathbf{C} be a class of graphs. Then the induced subgraph problem on \mathbf{C} is

STREMB(\mathbf{C})

Input: Two graphs G and H with $G \in \mathbf{C}$.

Problem: Does H contain an induced subgraph isomorphic to G ?

\mathbf{C} is the class of needles, and a haystack is a graph H .

Proviso: We always assume that \mathbf{C} is recursively enumerable: *there is an algorithm to enumerate all elements in \mathbf{C} .*

Induced subgraph problems (cont'd)

Induced subgraph problems (cont'd)

Definition

Let \mathbf{C} be a class of graphs. Then the counting induced subgraph problem on \mathbf{C} is

$\#STREMB(\mathbf{C})$

Input: Two graphs G and H with $G \in \mathbf{C}$.

Problem: Count the **number** of induced subgraphs in H isomorphic to G ?

Induced subgraph problems (cont'd)

Definition

Let \mathbf{C} be a class of graphs. Then the counting induced subgraph problem on \mathbf{C} is

$\#\text{STREMB}(\mathbf{C})$

Input: Two graphs G and H with $G \in \mathbf{C}$.

Problem: Count the **number** of induced subgraphs in H isomorphic to G ?

Obviously, $\#\text{STREMB}(\mathbf{C})$ is *at least as hard as* $\text{STREMB}(\mathbf{C})$.

Clique problem

Clique problem

If we take

$$\mathbf{C} := \{K_k \mid k \in \mathbb{N}\}$$

where K_k is the **complete graph** on k vertices, then $\text{STREMB}(\mathbf{C})$ is (equivalent to) the classical **clique problem**:

Clique problem

If we take

$$\mathbf{C} := \{K_k \mid k \in \mathbb{N}\}$$

where K_k is the **complete graph** on k vertices, then $\text{STREMB}(\mathbf{C})$ is (equivalent to) the classical **clique problem**:

CLIQUE

Input: A graph G and a natural number k .

Problem: Does G contain **k vertices** such that **every pair of them are adjacent?**

Clique problem

If we take

$$\mathbf{C} := \{K_k \mid k \in \mathbb{N}\}$$

where K_k is the **complete graph** on k vertices, then $\text{STREMB}(\mathbf{C})$ is (equivalent to) the classical **clique problem**:

CLIQUE

Input: A graph G and a natural number k .

Problem: Does G contain **k vertices** such that **every pair of them are adjacent?**

Theorem (Karp, 1976)

CLIQUE is **NP-complete**.

Chordless Path problem

Chordless Path problem

If we take

$$\mathbf{C} := \{P_k \mid k \in \mathbb{N}\}$$

where P_k is the **generic simple path** of k vertices, then $\text{STREMB}(\mathbf{C})$ is (equivalent to) the classical **chordless path problem**:

Chordless Path problem

If we take

$$\mathbf{C} := \{P_k \mid k \in \mathbb{N}\}$$

where P_k is the **generic simple path** of k vertices, then $\text{STREMB}(\mathbf{C})$ is (equivalent to) the classical **chordless path problem**:

CHORDLESS-PATH

Input: A graph G and a natural number k .

Problem: Does G contain an **induced path** of length k ?

Chordless Path problem

If we take

$$\mathbf{C} := \{P_k \mid k \in \mathbb{N}\}$$

where P_k is the **generic simple path** of k vertices, then $\text{STREMB}(\mathbf{C})$ is (equivalent to) the classical **chordless path problem**:

CHORDLESS-PATH

Input: A graph G and a natural number k .

Problem: Does G contain an **induced path** of length k ?

Theorem (Yannakakis, 1978)

CHORDLESS-PATH is **NP-complete**.

The hardness of induced subgraph problems

The hardness of induced subgraph problems

If \mathbf{C} is finite, then clearly $\text{STREMB}(\mathbf{C})$ can be solved in polynomial time.

The hardness of induced subgraph problems

If \mathbf{C} is finite, then clearly $\text{STREMB}(\mathbf{C})$ can be solved in polynomial time.

Given the **NP**-hardness of the clique problem, the following conjecture seems reasonable:

$\text{STREMB}(\mathbf{C})$ is solvable in polynomial time if and only if \mathbf{C} is finite.

The rest of talk

The classical complexity of induced subgraph isomorphisms

Parameterized Complexity

The parameterized hardness of induced subgraph problems

- Some combinatorics

- The hardness of counting

- The hardness of decision

Conclusion and open problems

The classical complexity of induced subgraph isomorphisms

Conjecture *If \mathbf{C} is infinite, then $\text{STREMB}(\mathbf{C})$ is not solvable in polynomial time.*

Conjecture *If \mathbf{C} is infinite, then $\text{STREMB}(\mathbf{C})$ is not solvable in polynomial time.*

It implies $\mathbf{NP} \neq \mathbf{P}$.

Conjecture *If \mathbf{C} is infinite, then $\text{STREMB}(\mathbf{C})$ is not solvable in polynomial time.*

It implies $\mathbf{NP} \neq \mathbf{P}$.

Conjecture *If \mathbf{C} is infinite, then $\text{STREMB}(\mathbf{C})$ is \mathbf{NP} -hard.*

The structure within **NP**

The structure within **NP**

Theorem (Ladner, 1975)

*Assume **NP** \neq **P**. Then there is a problem that is neither decidable in polynomial time nor **NP**-hard.*

The structure within **NP**

Theorem (Ladner, 1975)

*Assume **NP** \neq **P**. Then there is a problem that is neither decidable in polynomial time nor **NP**-hard.*

Theorem (Schaefer, 1978)

*Every **Boolean Constraint Satisfaction** problem is either decidable in polynomial time or **NP**-hard.*

Theorem (C. , Thurley, and Weyer, 2008)

- ▶ *There exists some (reasonable) \mathbf{C} such that $\text{STREMB}(\mathbf{C})$ is neither decidable in polynomial time nor \mathbf{NP} -hard, unless $\mathbf{NP} = \mathbf{P}$.*

Theorem (C. , Thurley, and Weyer, 2008)

- ▶ *There exists some (reasonable) \mathbf{C} such that $\text{STREMB}(\mathbf{C})$ is neither decidable in polynomial time nor \mathbf{NP} -hard, unless $\mathbf{NP} = \mathbf{P}$.*
- ▶ *There exists some (reasonable) \mathbf{C} such that $\#\text{STREMB}(\mathbf{C})$ is neither decidable in polynomial time nor $\#\mathbf{P}$ -hard, unless $\#\mathbf{P} = \mathbf{P}$.*

No finite classification in classical complexity

Theorem (induced subgraph problems are **dense**)

No finite classification in classical complexity

Theorem (induced subgraph problems are **dense**)

For every *polynomial time decidable* \mathbf{C}_1 and \mathbf{C}_2 with

$$\text{STREMB}(\mathbf{C}_1) < \text{STREMB}(\mathbf{C}_2),$$

No finite classification in classical complexity

Theorem (induced subgraph problems are **dense**)

For every *polynomial time decidable* \mathbf{C}_1 and \mathbf{C}_2 with

$$\text{STREMB}(\mathbf{C}_1) < \text{STREMB}(\mathbf{C}_2),$$

there exists some polynomial time decidable \mathbf{C} such that

$$\text{STREMB}(\mathbf{C}_1) < \text{STREMB}(\mathbf{C}) < \text{STREMB}(\mathbf{C}_2).$$

No finite classification in classical complexity

Theorem (induced subgraph problems are **dense**)

For every **polynomial time decidable** \mathbf{C}_1 and \mathbf{C}_2 with

$$\text{STREMB}(\mathbf{C}_1) < \text{STREMB}(\mathbf{C}_2),$$

there exists some polynomial time decidable \mathbf{C} such that

$$\text{STREMB}(\mathbf{C}_1) < \text{STREMB}(\mathbf{C}) < \text{STREMB}(\mathbf{C}_2).$$

$Q < Q'$: there is a polynomial time reduction from Q to Q' , but there is **no** polynomial time reduction from Q' to Q .

Proof. [Slow Diagonalization]

Proof. [Slow Diagonalization]

We aim to define a polynomial time computable function $f : \mathbb{N} \rightarrow \mathbb{N}$ with input in unary.

Proof. [Slow Diagonalization]

We aim to define a polynomial time computable function $f : \mathbb{N} \rightarrow \mathbb{N}$ with input in unary.

Then let

$$\mathbf{C} := \{G \mid G \in \mathbf{C}_1 \text{ or } (G \in \mathbf{C}_2 \text{ and } f(|V(G)|) \text{ is even})\}.$$

Proof. [Slow Diagonalization]

We aim to define a polynomial time computable function $f : \mathbb{N} \rightarrow \mathbb{N}$ with input in unary.

Then let

$$\mathbf{C} := \{G \mid G \in \mathbf{C}_1 \text{ or } (G \in \mathbf{C}_2 \text{ and } f(|V(G)|) \text{ is even})\}.$$

Thus,

$$\text{STREMB}(\mathbf{C}_1) \leq \text{STREMB}(\mathbf{C}) \leq \text{STREMB}(\mathbf{C}_2).$$

Proof. [Slow Diagonalization]

We aim to define a polynomial time computable function $f : \mathbb{N} \rightarrow \mathbb{N}$ with input in unary.

Then let

$$\mathbf{C} := \{G \mid G \in \mathbf{C}_1 \text{ or } (G \in \mathbf{C}_2 \text{ and } f(|V(G)|) \text{ is even})\}.$$

Thus,

$$\text{STREMB}(\mathbf{C}_1) \leq \text{STREMB}(\mathbf{C}) \leq \text{STREMB}(\mathbf{C}_2).$$

$Q \leq Q'$: *there is a polynomial time reduction from Q to Q' .*

Proof. (cont'd)

Proof. (cont'd)

So the main problem is to define f in such that way that

$$\text{STREMB}(\mathbf{C}_2) \not\leq \text{STREMB}(\mathbf{C}) \not\leq \text{STREMB}(\mathbf{C}_1).$$

Proof. (cont'd)

So the main problem is to define f in such that way that

$$\text{STREMB}(\mathbf{C}_2) \not\leq \text{STREMB}(\mathbf{C}) \not\leq \text{STREMB}(\mathbf{C}_1).$$

R_1, R_2, \dots : an **effective enumeration** of all polynomial time reductions.

Proof. (cont'd)

So the main problem is to define f in such that way that

$$\text{STREMB}(\mathbf{C}_2) \not\leq \text{STREMB}(\mathbf{C}) \not\leq \text{STREMB}(\mathbf{C}_1).$$

R_1, R_2, \dots : an **effective enumeration** of all polynomial time reductions.

For each $i \in \mathbb{N}$:

Proof. (cont'd)

So the main problem is to define f in such that way that

$$\text{STREMB}(\mathbf{C}_2) \not\subseteq \text{STREMB}(\mathbf{C}) \not\subseteq \text{STREMB}(\mathbf{C}_1).$$

R_1, R_2, \dots : an **effective enumeration** of all polynomial time reductions.

For each $i \in \mathbb{N}$:

S_i : For some graphs G and H with $G \in \mathbf{C}_2$,
 $(G, H) \in \text{STREMB}(\mathbf{C}_2) \iff R_i(G, H) \notin \text{STREMB}(\mathbf{C}).$

Proof. (cont'd)

So the main problem is to define f in such that way that

$$\text{STREMB}(\mathbf{C}_2) \not\leq \text{STREMB}(\mathbf{C}) \not\leq \text{STREMB}(\mathbf{C}_1).$$

R_1, R_2, \dots : an **effective enumeration** of all polynomial time reductions.

For each $i \in \mathbb{N}$:

- S_i : For some graphs G and H with $G \in \mathbf{C}_2$,
 $(G, H) \in \text{STREMB}(\mathbf{C}_2) \iff R_i(G, H) \notin \text{STREMB}(\mathbf{C})$.
That is, R_i is not a polynomial time reduction from
 $\text{STREMB}(\mathbf{C}_2)$ to $\text{STREMB}(\mathbf{C})$.

Proof. (cont'd)

So the main problem is to define f in such that way that

$$\text{STREMB}(\mathbf{C}_2) \not\leq \text{STREMB}(\mathbf{C}) \not\leq \text{STREMB}(\mathbf{C}_1).$$

R_1, R_2, \dots : an **effective enumeration** of all polynomial time reductions.

For each $i \in \mathbb{N}$:

S_i : For some graphs G and H with $G \in \mathbf{C}_2$,
 $(G, H) \in \text{STREMB}(\mathbf{C}_2) \iff R_i(G, H) \notin \text{STREMB}(\mathbf{C})$.
That is, R_i is not a polynomial time reduction from
 $\text{STREMB}(\mathbf{C}_2)$ to $\text{STREMB}(\mathbf{C})$.

N_i : For some graphs G and H with $G \in \mathbf{C}$,
 $(G, H) \in \text{STREMB}(\mathbf{C}) \iff R_i(G, H) \notin \text{STREMB}(\mathbf{C}_1)$.

Proof. (cont'd)

So the main problem is to define f in such that way that

$$\text{STREMB}(\mathbf{C}_2) \not\leq \text{STREMB}(\mathbf{C}) \not\leq \text{STREMB}(\mathbf{C}_1).$$

R_1, R_2, \dots : an **effective enumeration** of all polynomial time reductions.

For each $i \in \mathbb{N}$:

S_i : For some graphs G and H with $G \in \mathbf{C}_2$,
 $(G, H) \in \text{STREMB}(\mathbf{C}_2) \iff R_i(G, H) \notin \text{STREMB}(\mathbf{C})$.
That is, R_i is not a polynomial time reduction from
 $\text{STREMB}(\mathbf{C}_2)$ to $\text{STREMB}(\mathbf{C})$.

N_i : For some graphs G and H with $G \in \mathbf{C}$,
 $(G, H) \in \text{STREMB}(\mathbf{C}) \iff R_i(G, H) \notin \text{STREMB}(\mathbf{C}_1)$.
That is, R_i is not a polynomial time reduction from
 $\text{STREMB}(\mathbf{C})$ to $\text{STREMB}(\mathbf{C}_1)$.

Proof. (cont'd)

Proof. (cont'd)

Recall we *want to* define

$$\mathbf{C} := \{G \mid G \in \mathbf{C}_1 \text{ or } (G \in \mathbf{C}_2 \text{ and } f(|V(G)|) \text{ is even})\}.$$

Proof. (cont'd)

Recall we *want to* define

$$\mathbf{C} := \{G \mid G \in \mathbf{C}_1 \text{ or } (G \in \mathbf{C}_2 \text{ and } f(|V(G)|) \text{ is even})\}.$$

Let $f(1) := 1$.

Proof. (cont'd)

Recall we *want to* define

$$\mathbf{C} := \{G \mid G \in \mathbf{C}_1 \text{ or } (G \in \mathbf{C}_2 \text{ and } f(|V(G)|) \text{ is even})\}.$$

Let $f(1) := 1$.

Now assume all $f(i)$ are defined for $i \leq n$, and we want to define $f(n+1)$.

Proof. (cont'd)

Recall we *want to* define

$$\mathbf{C} := \{G \mid G \in \mathbf{C}_1 \text{ or } (G \in \mathbf{C}_2 \text{ and } f(|V(G)|) \text{ is even})\}.$$

Let $f(1) := 1$.

Now assume all $f(i)$ are defined for $i \leq n$, and we want to define $f(n+1)$.

Thus we *already know*

$$\mathbf{C}^n := \{G \mid G \in \mathbf{C}_1 \text{ or } (G \in \mathbf{C}_2 \text{ with } |V(G)| \leq n \text{ and } f(|V(G)|) \text{ even})\}.$$

Proof. (cont'd)

Proof. (cont'd)

▶ $f(n) = 2i - 1$:

Proof. (cont'd)

- ▶ $f(n) = 2i - 1$: By *exhaustive search*, we are looking for a pair of graphs (G, H) with $G \in \mathbf{C}_2$, such that $(G, H) \in \text{STREMB}(\mathbf{C}_2) \iff R_i(G, H) \notin \text{STREMB}(\mathbf{C}^n)$.

Proof. (cont'd)

- ▶ $f(n) = 2i - 1$: By *exhaustive search*, we are looking for a pair of graphs (G, H) with $G \in \mathbf{C}_2$, such that $(G, H) \in \text{STREMB}(\mathbf{C}_2) \iff R_i(G, H) \notin \text{STREMB}(\mathbf{C}^n)$.
- ▶ $f(n) = 2i$: By *exhaustive search*, we are looking for a pair of graphs (G, H) with $G \in \mathbf{C}^n$, such that $(G, H) \in \text{STREMB}(\mathbf{C}^n) \iff R_i(G, H) \notin \text{STREMB}(\mathbf{C}_1)$.

Proof. (cont'd)

- ▶ $f(n) = 2i - 1$: By *exhaustive search*, we are looking for a pair of graphs (G, H) with $G \in \mathbf{C}_2$, such that $(G, H) \in \text{STREMB}(\mathbf{C}_2) \iff R_i(G, H) \notin \text{STREMB}(\mathbf{C}^n)$.
- ▶ $f(n) = 2i$: By *exhaustive search*, we are looking for a pair of graphs (G, H) with $G \in \mathbf{C}^n$, such that $(G, H) \in \text{STREMB}(\mathbf{C}^n) \iff R_i(G, H) \notin \text{STREMB}(\mathbf{C}_1)$.

We *abort* this search *after time* n , if it was not successful until then.

Proof. (cont'd)

- ▶ $f(n) = 2i - 1$: By *exhaustive search*, we are looking for a pair of graphs (G, H) with $G \in \mathbf{C}_2$, such that $(G, H) \in \text{STREMB}(\mathbf{C}_2) \iff R_i(G, H) \notin \text{STREMB}(\mathbf{C}^n)$.
- ▶ $f(n) = 2i$: By *exhaustive search*, we are looking for a pair of graphs (G, H) with $G \in \mathbf{C}^n$, such that $(G, H) \in \text{STREMB}(\mathbf{C}^n) \iff R_i(G, H) \notin \text{STREMB}(\mathbf{C}_1)$.

We *abort* this search *after time* n , if it was not successful until then.

Note this time constraint implies that we will never “*meet*” instance $R_i(G, H)$ or (G, H) of $\text{STREMB}(\mathbf{C}^n)$ with $|V(G)| \geq n$.

Proof. (cont'd)

- ▶ $f(n) = 2i - 1$: By *exhaustive search*, we are looking for a pair of graphs (G, H) with $G \in \mathbf{C}_2$, such that $(G, H) \in \text{STREMB}(\mathbf{C}_2) \iff R_i(G, H) \notin \text{STREMB}(\mathbf{C}^n)$.
- ▶ $f(n) = 2i$: By *exhaustive search*, we are looking for a pair of graphs (G, H) with $G \in \mathbf{C}^n$, such that $(G, H) \in \text{STREMB}(\mathbf{C}^n) \iff R_i(G, H) \notin \text{STREMB}(\mathbf{C}_1)$.

We *abort* this search *after time* n , if it was not successful until then.

Note this time constraint implies that we will never “*meet*” instance $R_i(G, H)$ or (G, H) of $\text{STREMB}(\mathbf{C}^n)$ with $|V(G)| \geq n$. If the search was *successful*, we let $f(n+1) := f(n) + 1$,

Proof. (cont'd)

- ▶ $f(n) = 2i - 1$: By *exhaustive search*, we are looking for a pair of graphs (G, H) with $G \in \mathbf{C}_2$, such that $(G, H) \in \text{STREMB}(\mathbf{C}_2) \iff R_i(G, H) \notin \text{STREMB}(\mathbf{C}^n)$.
- ▶ $f(n) = 2i$: By *exhaustive search*, we are looking for a pair of graphs (G, H) with $G \in \mathbf{C}^n$, such that $(G, H) \in \text{STREMB}(\mathbf{C}^n) \iff R_i(G, H) \notin \text{STREMB}(\mathbf{C}_1)$.

We *abort* this search *after time* n , if it was not successful until then.

Note this time constraint implies that we will never “*meet*” instance $R_i(G, H)$ or (G, H) of $\text{STREMB}(\mathbf{C}^n)$ with $|V(G)| \geq n$. If the search was *successful*, we let $f(n+1) := f(n) + 1$, otherwise $f(n+1) = f(n)$.

Proof. (cont'd)

- ▶ $f(n) = 2i - 1$: By *exhaustive search*, we are looking for a pair of graphs (G, H) with $G \in \mathbf{C}_2$, such that $(G, H) \in \text{STREMB}(\mathbf{C}_2) \iff R_i(G, H) \notin \text{STREMB}(\mathbf{C}^n)$.
- ▶ $f(n) = 2i$: By *exhaustive search*, we are looking for a pair of graphs (G, H) with $G \in \mathbf{C}^n$, such that $(G, H) \in \text{STREMB}(\mathbf{C}^n) \iff R_i(G, H) \notin \text{STREMB}(\mathbf{C}_1)$.

We *abort* this search *after time* n , if it was not successful until then.

Note this time constraint implies that we will never “meet” instance $R_i(G, H)$ or (G, H) of $\text{STREMB}(\mathbf{C}^n)$ with $|V(G)| \geq n$. If the search was *successful*, we let $f(n+1) := f(n) + 1$, otherwise $f(n+1) = f(n)$. $f(n+1)$ can be computed in *linear time*, if \mathbf{C}^n and $f(n)$ are known.

Proof. (cont'd)

- ▶ $f(n) = 2i - 1$: By *exhaustive search*, we are looking for a pair of graphs (G, H) with $G \in \mathbf{C}_2$, such that $(G, H) \in \text{STREMB}(\mathbf{C}_2) \iff R_i(G, H) \notin \text{STREMB}(\mathbf{C}^n)$.
- ▶ $f(n) = 2i$: By *exhaustive search*, we are looking for a pair of graphs (G, H) with $G \in \mathbf{C}^n$, such that $(G, H) \in \text{STREMB}(\mathbf{C}^n) \iff R_i(G, H) \notin \text{STREMB}(\mathbf{C}_1)$.

We *abort* this search *after time* n , if it was not successful until then.

Note this time constraint implies that we will never “meet” instance $R_i(G, H)$ or (G, H) of $\text{STREMB}(\mathbf{C}^n)$ with $|V(G)| \geq n$. If the search was *successful*, we let $f(n+1) := f(n) + 1$, otherwise $f(n+1) = f(n)$. $f(n+1)$ can be computed in *linear time*, if \mathbf{C}^n and $f(n)$ are known.

So, overall, f is computable in polynomial time.

Proof. (cont'd)

Proof. (cont'd)

We first prove that f is unbounded:

Proof. (cont'd)

We first prove that f is unbounded:

Otherwise, let n be such that

$$f(n) = \max\{f(m) \mid m \in \mathbb{N}\}.$$

Proof. (cont'd)

We first prove that f is unbounded:

Otherwise, let n be such that

$$f(n) = \max\{f(m) \mid m \in \mathbb{N}\}.$$

- ▶ If $f(n) = 2i - 1$, then for every $m \geq n$ we have $\mathbf{C}^m = \mathbf{C}^n$.

Proof. (cont'd)

We first prove that f is unbounded:

Otherwise, let n be such that

$$f(n) = \max\{f(m) \mid m \in \mathbb{N}\}.$$

- ▶ If $f(n) = 2i - 1$, then for every $m \geq n$ we have $\mathbf{C}^m = \mathbf{C}^n$. For every instance (G, H) with $G \in \mathbf{C}_2$, for sufficiently large $m \geq n$ (depending on G, H), $R_i(G, H) = (G', H')$ with $|V(G')| \leq m$.

Proof. (cont'd)

We first prove that f is unbounded:

Otherwise, let n be such that

$$f(n) = \max\{f(m) \mid m \in \mathbb{N}\}.$$

- ▶ If $f(n) = 2i - 1$, then for every $m \geq n$ we have $\mathbf{C}^m = \mathbf{C}^n$. For every instance (G, H) with $G \in \mathbf{C}_2$, for sufficiently large $m \geq n$ (depending on G, H), $R_i(G, H) = (G', H')$ with $|V(G')| \leq m$. Hence

$$\begin{aligned} (G', H') \in \text{STREMB}(\mathbf{C}) &\iff (G', H') \in \text{STREMB}(\mathbf{C}^m) \\ &\iff (G', H') \in \text{STREMB}(\mathbf{C}^n) \\ &\iff (G, H) \in \text{STREMB}(\mathbf{C}_2). \end{aligned}$$

Proof. (cont'd)

We first prove that f is unbounded:

Otherwise, let n be such that

$$f(n) = \max\{f(m) \mid m \in \mathbb{N}\}.$$

- ▶ If $f(n) = 2i - 1$, then for every $m \geq n$ we have $\mathbf{C}^m = \mathbf{C}^n$. For every instance (G, H) with $G \in \mathbf{C}_2$, for sufficiently large $m \geq n$ (depending on G, H), $R_i(G, H) = (G', H')$ with $|V(G')| \leq m$. Hence

$$\begin{aligned} (G', H') \in \text{STREMB}(\mathbf{C}) &\iff (G', H') \in \text{STREMB}(\mathbf{C}^m) \\ &\iff (G', H') \in \text{STREMB}(\mathbf{C}^n) \\ &\iff (G, H) \in \text{STREMB}(\mathbf{C}_2). \end{aligned}$$

Thus $\text{STREMB}(\mathbf{C}_2) \leq \text{STREMB}(\mathbf{C})$.

Proof. (cont'd)

We first prove that f is unbounded:

Otherwise, let n be such that

$$f(n) = \max\{f(m) \mid m \in \mathbb{N}\}.$$

- ▶ If $f(n) = 2i - 1$, then for every $m \geq n$ we have $\mathbf{C}^m = \mathbf{C}^n$. For every instance (G, H) with $G \in \mathbf{C}_2$, for sufficiently large $m \geq n$ (depending on G, H), $R_i(G, H) = (G', H')$ with $|V(G')| \leq m$. Hence

$$\begin{aligned} (G', H') \in \text{STREMB}(\mathbf{C}) &\iff (G', H') \in \text{STREMB}(\mathbf{C}^m) \\ &\iff (G', H') \in \text{STREMB}(\mathbf{C}^n) \\ &\iff (G, H) \in \text{STREMB}(\mathbf{C}_2). \end{aligned}$$

Thus $\text{STREMB}(\mathbf{C}_2) \leq \text{STREMB}(\mathbf{C})$. As $\mathbf{C} \setminus \mathbf{C}_1$ is finite,

Proof. (cont'd)

We first prove that f is unbounded:

Otherwise, let n be such that

$$f(n) = \max\{f(m) \mid m \in \mathbb{N}\}.$$

- ▶ If $f(n) = 2i - 1$, then for every $m \geq n$ we have $\mathbf{C}^m = \mathbf{C}^n$. For every instance (G, H) with $G \in \mathbf{C}_2$, for sufficiently large $m \geq n$ (depending on G, H), $R_i(G, H) = (G', H')$ with $|V(G')| \leq m$. Hence

$$\begin{aligned}(G', H') \in \text{STREMB}(\mathbf{C}) &\iff (G', H') \in \text{STREMB}(\mathbf{C}^m) \\ &\iff (G', H') \in \text{STREMB}(\mathbf{C}^n) \\ &\iff (G, H) \in \text{STREMB}(\mathbf{C}_2).\end{aligned}$$

Thus $\text{STREMB}(\mathbf{C}_2) \leq \text{STREMB}(\mathbf{C})$. As $\mathbf{C} \setminus \mathbf{C}_1$ is finite,

$$\text{STREMB}(\mathbf{C}_2) \leq \text{STREMB}(\mathbf{C}_1),$$

a contradiction.

Proof. (cont'd)

Proof. (cont'd)

- ▶ If $f(n) = 2i$, then for every (G, H) with $G \in \mathbf{C}$

$$(G, H) \in \text{STREMB}(\mathbf{C}) \iff R_i(G, H) \in \text{STREMB}(\mathbf{C}_1).$$

Proof. (cont'd)

- ▶ If $f(n) = 2i$, then for every (G, H) with $G \in \mathbf{C}$

$$(G, H) \in \text{STREMB}(\mathbf{C}) \iff R_i(G, H) \in \text{STREMB}(\mathbf{C}_1).$$

As \mathbf{C} only differs from $\mathbf{C}_1 \cup \mathbf{C}_2$ by finitely many graphs, we have

$$\text{STREMB}(\mathbf{C}_1 \cup \mathbf{C}_2) \leq \text{STREMB}(\mathbf{C}_1)$$

Proof. (cont'd)

- ▶ If $f(n) = 2i$, then for every (G, H) with $G \in \mathbf{C}$

$$(G, H) \in \text{STREMB}(\mathbf{C}) \iff R_i(G, H) \in \text{STREMB}(\mathbf{C}_1).$$

As \mathbf{C} only differs from $\mathbf{C}_1 \cup \mathbf{C}_2$ by finitely many graphs, we have

$$\text{STREMB}(\mathbf{C}_1 \cup \mathbf{C}_2) \leq \text{STREMB}(\mathbf{C}_1)$$

which implies

$$\text{STREMB}(\mathbf{C}_2) \leq \text{STREMB}(\mathbf{C}_1)$$

again a contradiction.

Proof. (cont'd)

Proof. (cont'd)

Now we want to show that every condition S_i is satisfied:

Proof. (cont'd)

Now we want to show that every condition S_i is satisfied: there must exist an n such that $f(n) = 2i - 1$ and $f(n + 1) = 2i$.

Proof. (cont'd)

Now we want to show that every condition S_i is satisfied: there must exist an n such that $f(n) = 2i - 1$ and $f(n + 1) = 2i$. It implies that for some (G, H) with $G \in \mathbf{C}_2$ we have

$$(G, H) \in \text{STREMB}(\mathbf{C}_2) \iff R_i(G, H) \notin \text{STREMB}(\mathbf{C}^n)$$

Proof. (cont'd)

Now we want to show that every condition S_i is satisfied: there must exist an n such that $f(n) = 2i - 1$ and $f(n + 1) = 2i$. It implies that for some (G, H) with $G \in \mathbf{C}_2$ we have

$$(G, H) \in \text{STREMB}(\mathbf{C}_2) \iff R_i(G, H) \notin \text{STREMB}(\mathbf{C}^n)$$

But

$$R_i(G, H) \notin \text{STREMB}(\mathbf{C}^n) \iff R_i(G, H) \notin \text{STREMB}(\mathbf{C})$$

Proof. (cont'd)

Now we want to show that every condition S_i is satisfied: there must exist an n such that $f(n) = 2i - 1$ and $f(n + 1) = 2i$. It implies that for some (G, H) with $G \in \mathbf{C}_2$ we have

$$(G, H) \in \text{STREMB}(\mathbf{C}_2) \iff R_i(G, H) \notin \text{STREMB}(\mathbf{C}^n)$$

But

$$R_i(G, H) \notin \text{STREMB}(\mathbf{C}^n) \iff R_i(G, H) \notin \text{STREMB}(\mathbf{C})$$

Similarly every N_i is satisfied. □

Is a dichotomy theorem possible?

Our main result

Our main result

Theorem

- ▶ $\text{STREMB}(\mathbf{C})$ is decidable in polynomial time if and only if \mathbf{C} is finite,

Our main result

Theorem

- ▶ $\text{STREMB}(\mathbf{C})$ is decidable in polynomial time if and only if \mathbf{C} is finite, *unless* $\mathbf{W}[1] = \mathbf{FPT}$.

Our main result

Theorem

- ▶ $\text{STREMB}(\mathbf{C})$ is decidable in polynomial time if and only if \mathbf{C} is finite, *unless* $\mathbf{W}[1] = \mathbf{FPT}$.
- ▶ $\#\text{STREMB}(\mathbf{C})$ is decidable in polynomial time if and only if \mathbf{C} is finite,

Our main result

Theorem

- ▶ $\text{STREMB}(\mathbf{C})$ is decidable in polynomial time if and only if \mathbf{C} is finite, *unless* $\mathbf{W}[1] = \mathbf{FPT}$.
- ▶ $\#\text{STREMB}(\mathbf{C})$ is decidable in polynomial time if and only if \mathbf{C} is finite, *unless* $\#\mathbf{W}[1] = \mathbf{FPT}$.

Parameterized Complexity

Parameterized problems

Definition

A parameterized problem is a pair (Q, κ) where $Q \subseteq \Sigma^*$ is a classical problem and $\kappa : \Sigma^* \rightarrow \mathbb{N}$ is computable in polynomial time (i.e., a **parameterization** of Σ^*).

Parameterized problems

Definition

A parameterized problem is a pair (Q, κ) where $Q \subseteq \Sigma^*$ is a classical problem and $\kappa : \Sigma^* \rightarrow \mathbb{N}$ is computable in polynomial time (i.e., a **parameterization** of Σ^*).

Definition

A parameterized counting problem is a pair (F, κ) where $F : \Sigma^* \rightarrow \mathbb{N}$ and κ is a parameterization.

Parameterized clique problem

The parameterized clique problem

p -CLIQUE

Input: A graph $G = (V, E)$ and a natural number k .

Parameter: k .

Problem: Does G contain an induced subgraph isomorphic to K_k ?

Parameterized clique problem

The parameterized clique problem

p -CLIQUE

- Input:* A graph $G = (V, E)$ and a natural number k .
- Parameter:* k .
- Problem:* Does G contain an induced subgraph isomorphic to K_k ?

The counting version

p -#CLIQUE

- Input:* A graph $G = (V, E)$ and a natural number k .
- Parameter:* k .
- Problem:* Compute the **number** of induced subgraphs in G isomorphic to K_k .

Parameterized induced subgraph problems

Parameterized induced subgraph problems

Definition

Let \mathbf{C} be a class of graphs. Then the parameterized induced subgraph problem on \mathbf{C} is

p -STREMB(\mathbf{C})

Input: Two graphs G and H with $G \in \mathbf{C}$.

Parameter: $\|G\|$.

Problem: Does H contain an induced subgraph isomorphic to G ?

where $\|G\| = |V(G)| + |E(G)|$.

Parameterized induced subgraph problems

Definition

Let \mathbf{C} be a class of graphs. Then the parameterized induced subgraph problem on \mathbf{C} is

p -STREMB(\mathbf{C})

Input: Two graphs G and H with $G \in \mathbf{C}$.

Parameter: $\|G\|$.

Problem: Does H contain an induced subgraph isomorphic to G ?

where $\|G\| = |V(G)| + |E(G)|$. And the counting version:

p -#STREMB(\mathbf{C})

Input: Two graphs G and H with $G \in \mathbf{C}$.

Parameter: $\|G\|$.

Problem: Count the number of induced subgraphs in H isomorphic to G .

Fixed-parameter tractability

Definition

Let (Q, κ) be a parameterized problem. (Q, κ) is fixed-parameter tractable (or $(Q, \kappa) \in \mathbf{FPT}$) if there exists an algorithm \mathbb{A} that decides $x \in Q$ in time $f(\kappa(x)) \cdot |x|^{O(1)}$, where f is a **computable function**.

Fixed-parameter tractability

Definition

Let (Q, κ) be a parameterized problem. (Q, κ) is fixed-parameter tractable (or $(Q, \kappa) \in \mathbf{FPT}$) if there exists an algorithm \mathbb{A} that decides $x \in Q$ in time $f(\kappa(x)) \cdot |x|^{O(1)}$, where f is a **computable function**.

Definition

Let (F, κ) be a parameterized counting problem. (F, κ) is fixed-parameter tractable (or $(F, \kappa) \in \mathbf{FPT}$) if there exists an algorithm \mathbb{A} such that for every instance $x \in \Sigma^*$ the algorithm \mathbb{A} computes $F(x)$ in time $f(\kappa(x)) \cdot |x|^{O(1)}$, where f is a computable function.

The parameterized intractability

The parameterized intractability

Theorem (Downey and Fellows, 1995)

*If 3SAT is not decidable in **subexponential time**, then $p\text{-CLIQUE}$ is not fixed-parameter tractable.*

The parameterized intractability

Theorem (Downey and Fellows, 1995)

*If 3SAT is not decidable in **subexponential time**, then p -CLIQUE is not fixed-parameter tractable.*

Thus, p -#CLIQUE is not fixed-parameter tractable either.

The parameterized reductions

Definition (fpt many-one reduction between decision problems)

$(Q_1, \kappa_1) \leq^{\text{fpt}} (Q_2, \kappa_2)$ iff there is a **mapping** $R : \{0, 1\}^* \rightarrow \{0, 1\}^*$ such that:

The parameterized reductions

Definition (fpt many-one reduction between decision problems)

$(Q_1, \kappa_1) \leq^{\text{fpt}} (Q_2, \kappa_2)$ iff there is a mapping $R : \{0, 1\}^* \rightarrow \{0, 1\}^*$ such that:

1. $x \in Q_1 \iff R(x) \in Q_2$.

The parameterized reductions

Definition (fpt many-one reduction between decision problems)

$(Q_1, \kappa_1) \leq^{\text{fpt}} (Q_2, \kappa_2)$ iff there is a mapping $R : \{0, 1\}^* \rightarrow \{0, 1\}^*$ such that:

1. $x \in Q_1 \iff R(x) \in Q_2$.
2. $R(x)$ is computable in time

$$f(\kappa(x)) \cdot |x|^{O(1)}$$

for a computable $f : \mathbb{N} \rightarrow \mathbb{N}$.

The parameterized reductions

Definition (fpt many-one reduction between decision problems)

$(Q_1, \kappa_1) \leq^{\text{fpt}} (Q_2, \kappa_2)$ iff there is a mapping $R : \{0, 1\}^* \rightarrow \{0, 1\}^*$ such that:

1. $x \in Q_1 \iff R(x) \in Q_2$.
2. $R(x)$ is computable in time

$$f(\kappa(x)) \cdot |x|^{O(1)}$$

for a computable $f : \mathbb{N} \rightarrow \mathbb{N}$.

3. There is a computable function $g : \mathbb{N} \rightarrow \mathbb{N}$ such that

$$\kappa_2(R(x)) \leq g(\kappa_1(x)).$$

The parameterized reductions

Definition (fpt many-one reduction between decision problems)

$(Q_1, \kappa_1) \leq^{\text{fpt}} (Q_2, \kappa_2)$ iff there is a mapping $R : \{0, 1\}^* \rightarrow \{0, 1\}^*$ such that:

1. $x \in Q_1 \iff R(x) \in Q_2$.
2. $R(x)$ is computable in time

$$f(\kappa(x)) \cdot |x|^{O(1)}$$

for a computable $f : \mathbb{N} \rightarrow \mathbb{N}$.

3. There is a computable function $g : \mathbb{N} \rightarrow \mathbb{N}$ such that

$$\kappa_2(R(x)) \leq g(\kappa_1(x)).$$

Proposition If $(Q_1, \kappa_1) \leq^{\text{fpt}} (Q_2, \kappa_2)$ and $(Q_2, \kappa_2) \in \mathbf{FPT}$, then $(Q_1, \kappa_1) \in \mathbf{FPT}$.

The parameterized reductions (cont'd)

Definition (fpt Turing reduction between counting problems)

$(F_1, \kappa_1) \leq^{\text{fpt-T}} (F_2, \kappa_2)$ iff there is an algorithm \mathbb{A} with an oracle to F_2 such that: on every input x

The parameterized reductions (cont'd)

Definition (fpt Turing reduction between counting problems)

$(F_1, \kappa_1) \leq^{\text{fpt-T}} (F_2, \kappa_2)$ iff there is an algorithm \mathbb{A} with an oracle to F_2 such that: on every input x

1. \mathbb{A} computes $F_1(x)$.

The parameterized reductions (cont'd)

Definition (fpt Turing reduction between counting problems)

$(F_1, \kappa_1) \leq^{\text{fpt-T}} (F_2, \kappa_2)$ iff there is an algorithm \mathbb{A} with an oracle to F_2 such that: on every input x

1. \mathbb{A} computes $F_1(x)$.
2. \mathbb{A} runs in time $f(\kappa_1(x)) \cdot |x|^{O(1)}$ for a computable function $f : \mathbb{N} \rightarrow \mathbb{N}$.

The parameterized reductions (cont'd)

Definition (fpt Turing reduction between counting problems)

$(F_1, \kappa_1) \leq^{\text{fpt-T}} (F_2, \kappa_2)$ iff there is an algorithm \mathbb{A} with an oracle to F_2 such that: on every input x

1. \mathbb{A} computes $F_1(x)$.
2. \mathbb{A} runs in time $f(\kappa_1(x)) \cdot |x|^{O(1)}$ for a computable function $f : \mathbb{N} \rightarrow \mathbb{N}$.
3. There is a computable function $g : \mathbb{N} \rightarrow \mathbb{N}$ such that for all oracle queries " $F_2(y) = ?$ " posed by \mathbb{A} we have

$$\kappa_2(y) \leq g(\kappa_1(x)).$$

The parameterized reductions (cont'd)

Definition (fpt Turing reduction between counting problems)

$(F_1, \kappa_1) \leq^{\text{fpt-T}} (F_2, \kappa_2)$ iff there is an algorithm \mathbb{A} with an oracle to F_2 such that: on every input x

1. \mathbb{A} computes $F_1(x)$.
2. \mathbb{A} runs in time $f(\kappa_1(x)) \cdot |x|^{O(1)}$ for a computable function $f : \mathbb{N} \rightarrow \mathbb{N}$.
3. There is a computable function $g : \mathbb{N} \rightarrow \mathbb{N}$ such that for all oracle queries " $F_2(y) = ?$ " posed by \mathbb{A} we have

$$\kappa_2(y) \leq g(\kappa_1(x)).$$

Proposition If $(F_1, \kappa_1) \leq^{\text{fpt-T}} (F_2, \kappa_2)$ and $(F_2, \kappa_2) \in \mathbf{FPT}$, then $(F_1, \kappa_1) \in \mathbf{FPT}$.

The classes $\mathbf{W}[1]$ and $\#\mathbf{W}[1]$

Definition

$$\mathbf{W}[1] := \{(Q, \kappa) \mid (Q, \kappa) \leq^{\text{fpt}} p\text{-CLIQUE}\}.$$

The classes $\mathbf{W[1]}$ and $\#\mathbf{W[1]}$

Definition

$$\mathbf{W[1]} := \{(Q, \kappa) \mid (Q, \kappa) \leq^{\text{fpt}} \mathbf{p}\text{-CLIQUE}\}.$$

Definition

$$\#\mathbf{W[1]} := \{(F, \kappa) \mid (F, \kappa) \leq^{\text{fpt-T}} \mathbf{p}\text{-}\#\text{CLIQUE}\}.$$

The classes $\mathbf{W}[1]$ and $\#\mathbf{W}[1]$

Definition

$$\mathbf{W}[1] := \{(Q, \kappa) \mid (Q, \kappa) \leq^{\text{fpt}} \rho\text{-CLIQUE}\}.$$

Definition

$$\#\mathbf{W}[1] := \{(F, \kappa) \mid (F, \kappa) \leq^{\text{fpt-T}} \rho\text{-}\#\text{CLIQUE}\}.$$

In the literature $\#\mathbf{W}[1]$ is defined via **parsimonious** reductions.

The classes $\mathbf{W[1]}$ and $\#\mathbf{W[1]}$ (cont'd)

Theorem (Downey and Fellows, 1995)

If 3SAT is not decidable in *subexponential time*, then

$$\mathbf{FPT} \subsetneq \mathbf{W[1]} \quad \text{and} \quad \mathbf{FPT} \subsetneq \#\mathbf{W[1]}.$$

The parameterized hardness of induced subgraph problems

Theorem (C. , Thurley, and Weyer, 2008)

- ▶ p -STREMB(\mathbf{C}) is **W[1]**-hard if \mathbf{C} is infinite.

The parameterized hardness of induced subgraph problems

Theorem (C. , Thurley, and Weyer, 2008)

- ▶ p -STREMB(\mathbf{C}) is $\mathbf{W}[1]$ -hard if \mathbf{C} is infinite.
- ▶ p -#STREMB(\mathbf{C}) is # $\mathbf{W}[1]$ -hard if \mathbf{C} is infinite.

The parameterized hardness of induced subgraph problems

Theorem (C. , Thurley, and Weyer, 2008)

- ▶ p -STREMB(\mathbf{C}) is $\mathbf{W}[1]$ -hard if \mathbf{C} is infinite.
- ▶ p -#STREMB(\mathbf{C}) is # $\mathbf{W}[1]$ -hard if \mathbf{C} is infinite.

Another formulation:

Theorem

If \mathbf{C} is infinite, then:

The parameterized hardness of induced subgraph problems

Theorem (C. , Thurley, and Weyer, 2008)

- ▶ p -STREMB(\mathbf{C}) is $\mathbf{W}[1]$ -hard if \mathbf{C} is infinite.
- ▶ p -#STREMB(\mathbf{C}) is # $\mathbf{W}[1]$ -hard if \mathbf{C} is infinite.

Another formulation:

Theorem

If \mathbf{C} is infinite, then:

- ▶ p -CLIQUE \leq^{fpt} p -STREMB(\mathbf{C});

The parameterized hardness of induced subgraph problems

Theorem (C. , Thurley, and Weyer, 2008)

- ▶ p -STREMB(\mathbf{C}) is $\mathbf{W}[1]$ -hard if \mathbf{C} is infinite.
- ▶ p -#STREMB(\mathbf{C}) is # $\mathbf{W}[1]$ -hard if \mathbf{C} is infinite.

Another formulation:

Theorem

If \mathbf{C} is infinite, then:

- ▶ p -CLIQUE \leq^{fpt} p -STREMB(\mathbf{C});
- ▶ p -#CLIQUE $\leq^{\text{fpt-T}}$ p -#STREMB(\mathbf{C}).

Ramsey's Theorem

Theorem

For every $k \in \mathbb{N}$ and every graph $G = (V, E)$, if $|V| \geq 2^{2^k}$, then G contains an induced subgraph either isomorphic to K_k (i.e., a clique of size k) or its **complement** (i.e., an independent set of size k).

Ramsey's Theorem

Theorem

For every $k \in \mathbb{N}$ and every graph $G = (V, E)$, if $|V| \geq 2^{2^k}$, then G contains an induced subgraph either isomorphic to K_k (i.e., a clique of size k) or its **complement** (i.e., an independent set of size k).

Corollary

Let \mathbf{C} be an **infinite** class of graphs. Then for every k , there exists a graph $G \in \mathbf{C}$ such that G contains either a clique of size k or an independent of size k .

Ramsey's Theorem

Theorem

For every $k \in \mathbb{N}$ and every graph $G = (V, E)$, if $|V| \geq 2^{2^k}$, then G contains an induced subgraph either isomorphic to K_k (i.e., a clique of size k) or its **complement** (i.e., an independent set of size k).

Corollary

Let \mathbf{C} be an **infinite** class of graphs. Then for every k , there exists a graph $G \in \mathbf{C}$ such that G contains either a clique of size k or an independent of size k .

If \mathbf{C} is **recursively enumerable**, then such G can be found effectively.

Graph complements

Definition

Let $G = (V, E)$ be a graph. Then its complement graph $G^{\text{comp}} = (V^{\text{comp}}, E^{\text{comp}})$ is defined by

$$V^{\text{comp}} := V$$

$$E^{\text{comp}} := \{ \{u, v\} \mid u, v \in V \text{ with } u \neq v \text{ and } \{u, v\} \notin E \}$$

Graph complements

Definition

Let $G = (V, E)$ be a graph. Then its complement graph $G^{\text{comp}} = (V^{\text{comp}}, E^{\text{comp}})$ is defined by

$$V^{\text{comp}} := V$$

$$E^{\text{comp}} := \{ \{u, v\} \mid u, v \in V \text{ with } u \neq v \text{ and } \{u, v\} \notin E \}$$

Lemma

G is an induced subgraph of H if and only if G^{comp} is an induced subgraph of H^{comp} .

Graph complements

Definition

Let $G = (V, E)$ be a graph. Then its complement graph $G^{\text{comp}} = (V^{\text{comp}}, E^{\text{comp}})$ is defined by

$$V^{\text{comp}} := V$$

$$E^{\text{comp}} := \{ \{u, v\} \mid u, v \in V \text{ with } u \neq v \text{ and } \{u, v\} \notin E \}$$

Lemma

G is an induced subgraph of H if and only if G^{comp} is an induced subgraph of H^{comp} .

Definition

Let \mathbf{C} be a class of graphs. Then its complement closure

$$\mathbf{C}^{\text{cc}} := \mathbf{C} \cup \{G^{\text{comp}} \mid G \in \mathbf{C}\}.$$

Graph complements (cont'd)

Lemma

- ▶ *If \mathbf{C} is infinite, then for every $k \in \mathbb{N}$, \mathbf{C}^{cc} contains a graph G such that G contains a clique of size k .*

Graph complements (cont'd)

Lemma

- ▶ If \mathbf{C} is infinite, then for every $k \in \mathbb{N}$, \mathbf{C}^{cc} contains a graph G such that G contains a clique of size k .
- ▶ $p\text{-STREMB}(\mathbf{C}^{\text{cc}}) \leq^{\text{fpt}} p\text{-STREMB}(\mathbf{C})$.

Graph complements (cont'd)

Lemma

- ▶ If \mathbf{C} is infinite, then for every $k \in \mathbb{N}$, \mathbf{C}^{cc} contains a graph G such that G contains a clique of size k .
- ▶ $p\text{-STREMB}(\mathbf{C}^{\text{cc}}) \leq^{\text{fpt}} p\text{-STREMB}(\mathbf{C})$.
- ▶ $p\text{-\#STREMB}(\mathbf{C}^{\text{cc}}) \leq^{\text{fpt-T}} p\text{-\#STREMB}(\mathbf{C})$.

Graph complements (cont'd)

Lemma

- ▶ If \mathbf{C} is infinite, then for every $k \in \mathbb{N}$, \mathbf{C}^{cc} contains a graph G such that G contains a clique of size k .
- ▶ $p\text{-STREMB}(\mathbf{C}^{\text{cc}}) \leq^{\text{fpt}} p\text{-STREMB}(\mathbf{C})$.
- ▶ $p\text{-}\#\text{STREMB}(\mathbf{C}^{\text{cc}}) \leq^{\text{fpt-T}} p\text{-}\#\text{STREMB}(\mathbf{C})$.

Thus, we can always assume that for every k , there exists a graph $G \in \mathbf{C}$ such that G contains a clique of size k .

The main reduction

The main reduction

Proposition *Let \mathbf{C} be a class of graphs such that for every k , there exists a graph $H \in \mathbf{C}$ such that H contains a clique of size k . Then $p\text{-}\#\text{CLIQUE} \leq^{\text{fpt-T}} p\text{-}\#\text{STREMB}(\mathbf{C})$.*

The main reduction

Proposition *Let \mathbf{C} be a class of graphs such that for every k , there exists a graph $H \in \mathbf{C}$ such that H contains a clique of size k . Then $p\text{-}\#\text{CLIQUE} \leq^{\text{fpt-T}} p\text{-}\#\text{STREMB}(\mathbf{C})$.*

Proof overview.

The main reduction

Proposition *Let \mathbf{C} be a class of graphs such that for every k , there exists a graph $H \in \mathbf{C}$ such that H contains a clique of size k . Then $p\text{-}\#\text{CLIQUE} \leq^{\text{fpt-T}} p\text{-}\#\text{STREMB}(\mathbf{C})$.*

Proof overview.

Let (G, k) be an instance of $p\text{-}\#\text{CLIQUE}$.

The main reduction

Proposition *Let \mathbf{C} be a class of graphs such that for every k , there exists a graph $H \in \mathbf{C}$ such that H contains a clique of size k . Then $p\text{-}\#\text{CLIQUE} \leq^{\text{fpt-T}} p\text{-}\#\text{STREMB}(\mathbf{C})$.*

Proof overview.

Let (G, k) be an instance of $p\text{-}\#\text{CLIQUE}$.

The reduction:

The main reduction

Proposition *Let \mathbf{C} be a class of graphs such that for every k , there exists a graph $H \in \mathbf{C}$ such that H contains a clique of size k . Then $p\text{-}\#\text{CLIQUE} \leq^{\text{fpt-T}} p\text{-}\#\text{STREMB}(\mathbf{C})$.*

Proof overview.

Let (G, k) be an instance of $p\text{-}\#\text{CLIQUE}$.

The reduction:

1. Find a graph $H \in \mathbf{C}$ such that H contains a clique of size k ,

The main reduction

Proposition *Let \mathbf{C} be a class of graphs such that for every k , there exists a graph $H \in \mathbf{C}$ such that H contains a clique of size k . Then $p\text{-}\#\text{CLIQUE} \leq^{\text{fpt-T}} p\text{-}\#\text{STREMB}(\mathbf{C})$.*

Proof overview.

Let (G, k) be an instance of $p\text{-}\#\text{CLIQUE}$.

The reduction:

1. Find a graph $H \in \mathbf{C}$ such that H contains a clique of size k , as \mathbf{C} is recursively enumerable.

The main reduction

Proposition *Let \mathbf{C} be a class of graphs such that for every k , there exists a graph $H \in \mathbf{C}$ such that H contains a clique of size k . Then $p\text{-}\#\text{CLIQUE} \leq^{\text{fpt-T}} p\text{-}\#\text{STREMB}(\mathbf{C})$.*

Proof overview.

Let (G, k) be an instance of $p\text{-}\#\text{CLIQUE}$.

The reduction:

1. Find a graph $H \in \mathbf{C}$ such that H contains a clique of size k , as \mathbf{C} is recursively enumerable.
2. Construct a graph H_G ,

The main reduction

Proposition *Let \mathbf{C} be a class of graphs such that for every k , there exists a graph $H \in \mathbf{C}$ such that H contains a clique of size k . Then $p\text{-}\#\text{CLIQUE} \leq^{\text{fpt-T}} p\text{-}\#\text{STREMB}(\mathbf{C})$.*

Proof overview.

Let (G, k) be an instance of $p\text{-}\#\text{CLIQUE}$.

The reduction:

1. Find a graph $H \in \mathbf{C}$ such that H contains a clique of size k , as \mathbf{C} is recursively enumerable.
2. Construct a graph H_G , and for each $I \subseteq [|V(H)|]$ a subgraph $H_{G,I}$ of H_G .

The main reduction

Proposition Let \mathbf{C} be a class of graphs such that for every k , there exists a graph $H \in \mathbf{C}$ such that H contains a clique of size k . Then $p\text{-}\#\text{CLIQUE} \leq^{\text{fpt-T}} p\text{-}\#\text{STREMB}(\mathbf{C})$.

Proof overview.

Let (G, k) be an instance of $p\text{-}\#\text{CLIQUE}$.

The reduction:

1. Find a graph $H \in \mathbf{C}$ such that H contains a clique of size k , as \mathbf{C} is recursively enumerable.
2. Construct a graph H_G , and for each $I \subseteq [|V(H)|]$ a subgraph $H_{G,I}$ of H_G .
3. Compute the number of k -cliques in G by asking the numbers of induced subgraphs in $H_{G,I}$ isomorphic to H .

The graph H_G

Let $k \in \mathbb{N}$ and $H = (V(H), E(H))$ be a graph with $V(H) = [n]$ such that $H \upharpoonright [k]$ is isomorphic to K_k .

The graph H_G

Let $k \in \mathbb{N}$ and $H = (V(H), E(H))$ be a graph with $V(H) = [n]$ such that $H \upharpoonright [k]$ is isomorphic to K_k .

Now let G be a graph. We construct a graph H_G with

$$V(H_G) := [k + 1, n] \cup V(G) \times [k]$$

$$E(H_G) := \left\{ \{i, j\} \mid i, j \in [k + 1, n] \text{ and } \{i, j\} \in E(H) \right\} \\ \cup \left\{ \{i, (v, j)\} \mid i \in [k + 1, n], v \in V(G), i \in [k], \right. \\ \quad \left. \text{and } \{i, j\} \in E(H) \right\} \\ \cup \left\{ \{(u, i), (v, j)\} \mid u, v \in V(G) \text{ with } \{u, v\} \in E(G), \right. \\ \quad \left. \text{and } i, j \in [k] \text{ with } i \neq j \right\}$$

Good induced subgraphs in H_G

Definition

An induce subgraph H' of H_G is good if

Good induced subgraphs in H_G

Definition

An induce subgraph H' of H_G is good if

- ▶ H' is isomorphic to H ;

Good induced subgraphs in H_G

Definition

An induce subgraph H' of H_G is good if

- ▶ H' is isomorphic to H ;
- ▶ for every $i \in [k]$ we have

$$|\{(u, i) \mid u \in V(G) \text{ and } (u, i) \in V(H')\}| = 1.$$

Good induced subgraphs in H_G

Definition

An induce subgraph H' of H_G is good if

- ▶ H' is isomorphic to H ;
- ▶ for every $i \in [k]$ we have

$$|\{(u, i) \mid u \in V(G) \text{ and } (u, i) \in V(H')\}| = 1.$$

Note, it implies that $[k + 1, n] \subseteq V(H')$.

Good induced subgraphs in H_G (cont'd)

Lemma

Let H' be a good induced subgraph of H_G , then

$$\{u \mid \text{for some } i \in [k] \text{ we have } (u, i) \in V(H')\}$$

is a clique of size k in G .

Good induced subgraphs in H_G (cont'd)

Lemma

Let H' be a good induced subgraph of H_G , then

$$\{u \mid \text{for some } i \in [k] \text{ we have } (u, i) \in V(H')\}$$

is a clique of size k in G .

Corollary

the number of good induced subgraphs in H_G
= the number of cliques of size k in G
 \times the number of automorphisms of $H \times k!$.

Counting the good induced subgraphs in H_G

Definition

Let $I \subseteq [n]$ (Recall $V(H) = [n]$). We define the graph $H_{G,I}$ as

$$H_G \upharpoonright ((I \cap [k+1, n]) \cup (V(G) \times (I \cap [k]))).$$

Counting the good induced subgraphs in H_G

Definition

Let $I \subseteq [n]$ (Recall $V(H) = [n]$). We define the graph $H_{G,I}$ as

$$H_G \upharpoonright ((I \cap [k+1, n]) \cup (V(G) \times (I \cap [k]))).$$

Note, $H_{G,[n]} = H_G$.

Counting the good induced subgraphs in H_G

Definition

Let $I \subseteq [n]$ (Recall $V(H) = [n]$). We define the graph $H_{G,I}$ as

$$H_G \upharpoonright ((I \cap [k+1, n]) \cup (V(G) \times (I \cap [k]))).$$

Note, $H_{G,[n]} = H_G$.

Definition

Let $I \subseteq [n]$. An induced subgraph H' in $H_{G,I}$ is I -good if

Counting the good induced subgraphs in H_G

Definition

Let $I \subseteq [n]$ (Recall $V(H) = [n]$). We define the graph $H_{G,I}$ as

$$H_G \upharpoonright ((I \cap [k+1, n]) \cup (V(G) \times (I \cap [k]))).$$

Note, $H_{G,[n]} = H_G$.

Definition

Let $I \subseteq [n]$. An induced subgraph H' in $H_{G,I}$ is I -good if

- ▶ H' is isomorphic to H ;

Counting the good induced subgraphs in H_G

Definition

Let $I \subseteq [n]$ (Recall $V(H) = [n]$). We define the graph $H_{G,I}$ as

$$H_G \upharpoonright ((I \cap [k+1, n]) \cup (V(G) \times (I \cap [k]))).$$

Note, $H_{G,[n]} = H_G$.

Definition

Let $I \subseteq [n]$. An induced subgraph H' in $H_{G,I}$ is I -good if

- ▶ H' is isomorphic to H ;
- ▶ for every $i \in [k]$ we have $V(H') \cap (V(G) \times \{i\}) \neq \emptyset$.

Counting the good induced subgraphs in H_G

Definition

Let $I \subseteq [n]$ (Recall $V(H) = [n]$). We define the graph $H_{G,I}$ as

$$H_G \upharpoonright ((I \cap [k+1, n]) \cup (V(G) \times (I \cap [k]))).$$

Note, $H_{G,[n]} = H_G$.

Definition

Let $I \subseteq [n]$. An induced subgraph H' in $H_{G,I}$ is I -good if

- ▶ H' is isomorphic to H ;
- ▶ for every $i \in [k]$ we have $V(H') \cap (V(G) \times \{i\}) \neq \emptyset$.
- ▶ $I \cap [k+1, n] = I \cap V(H')$.

Counting the good induced subgraphs in H_G

Definition

Let $I \subseteq [n]$ (Recall $V(H) = [n]$). We define the graph $H_{G,I}$ as

$$H_G \upharpoonright ((I \cap [k+1, n]) \cup (V(G) \times (I \cap [k]))).$$

Note, $H_{G,[n]} = H_G$.

Definition

Let $I \subseteq [n]$. An induced subgraph H' in $H_{G,I}$ is I -good if

- ▶ H' is isomorphic to H ;
- ▶ for every $i \in [k]$ we have $V(H') \cap (V(G) \times \{i\}) \neq \emptyset$.
- ▶ $I \cap [k+1, n] = I \cap V(H')$.

Note, H' is good in H_G if and H' is $[n]$ -good in $H_{G,[n]}$.

Counting by inclusion-exclusion

Definition

Let $I \subseteq [n]$.

$b_I :=$ the number of induced subgraphs in $H_{G,I}$ isomorphic to H ,

$c_I :=$ the number of I -good induced subgraphs in $H_{G,I}$.

Counting by inclusion-exclusion

Definition

Let $I \subseteq [n]$.

$b_I :=$ the number of induced subgraphs in $H_{G,I}$ isomorphic to H ,

$c_I :=$ the number of I -good induced subgraphs in $H_{G,I}$.

b_I is computable by the oracle to p -#STREMB(**C**).

Counting by inclusion-exclusion

Definition

Let $I \subseteq [n]$.

$b_I :=$ the number of induced subgraphs in $H_{G,I}$ isomorphic to H ,

$c_I :=$ the number of I -good induced subgraphs in $H_{G,I}$.

b_I is computable by the oracle to p -#STREMB(**C**).

Lemma (Inclusion and Exclusion)

For every $I \subseteq [n]$ we have

$$c_I = b_I - \sum_{I' \subsetneq I} c_{I'}.$$

Graph homomorphisms

Definition

Let G, H be two graphs. A mapping $h : V(G) \rightarrow V(H)$ is a homomorphism if for every $u, v \in V(G)$ we have

$$\{u, v\} \in E(G) \implies \{h(u), h(v)\} \in E(H).$$

Graph homomorphisms

Definition

Let G, H be two graphs. A mapping $h : V(G) \rightarrow V(H)$ is a homomorphism if for every $u, v \in V(G)$ we have

$$\{u, v\} \in E(G) \implies \{h(u), h(v)\} \in E(H).$$

Definition

Let \mathbf{C} be a class of graphs. Then the parameterized homomorphism problem on \mathbf{C}

p -HOM(\mathbf{C})

Input: Two graphs G and H with $G \in \mathbf{C}$.

Parameter: $\|G\|$.

Problem: Is there a homomorphism from G to H ?

Grohe's Theorem

Theorem

If for every $k \in \mathbb{N}$, there exists a graph $G \in \mathbf{C}$ such that G contains a clique of size k , then $p\text{-HOM}(\mathbf{C})$ is $\mathbf{W}[1]$ -hard.

From homomorphisms to induced subgraphs

Suggested by a reviewer:

Definition

Let G, H be two graphs. We define the product graph $G \times H$ by

$$V(G \times H) := V(G) \times V(H)$$

$$E(G \times H) := \left\{ \{(u_1, v_1), (u_2, v_2)\} \mid \{u_1, u_2\} \in E(G) \right. \\ \left. \text{and } \{u_2, v_2\} \in E(H) \right\}.$$

From homomorphisms to induced subgraphs

Suggested by a reviewer:

Definition

Let G, H be two graphs. We define the product graph $G \times H$ by

$$V(G \times H) := V(G) \times V(H)$$

$$E(G \times H) := \left\{ \{(u_1, v_1), (u_2, v_2)\} \mid \{u_1, u_2\} \in E(G) \right. \\ \left. \text{and } \{u_2, v_2\} \in E(H) \right\}.$$

Lemma

Let G, H be two graphs. There is a homomorphism from G to H if and only if $G \times H$ contains an induced subgraph isomorphic to G .

The hardness of induced subgraph

Corollary

Let \mathbf{C} be a class of graph such that for every $k \in \mathbb{N}$ there exists a graph $G \in \mathbf{C}$ such that G contains a clique of size k . Then $p\text{-STREMB}(\mathbf{C})$ is $\mathbf{W[1]}$ -hard.

The hardness of induced subgraph

Corollary

Let \mathbf{C} be a class of graph such that for every $k \in \mathbb{N}$ there exists a graph $G \in \mathbf{C}$ such that G contains a clique of size k . Then $p\text{-STREMB}(\mathbf{C})$ is $\mathbf{W[1]}$ -hard.

Theorem

Let \mathbf{C} be an infinite class of graphs. Then $p\text{-STREMB}(\mathbf{C})$ is $\mathbf{W[1]}$ -hard.

Conclusion and open problems

What are known now

What are known now

Assume $\mathbf{W[1]} \neq \mathbf{FPT}$:

What are known now

Assume $\mathbf{W}[1] \neq \mathbf{FPT}$:

Theorem

Let \mathbf{C} be a class of graphs.

What are known now

Assume $\mathbf{W}[1] \neq \mathbf{FPT}$:

Theorem

Let \mathbf{C} be a class of graphs.

- ▶ [Grohe, 2003] $\text{HOM}(\mathbf{C})$ is solvable in polynomial time if and only if *the cores of graphs in \mathbf{C} have bounded tree-width*.

What are known now

Assume $\mathbf{W}[1] \neq \mathbf{FPT}$:

Theorem

Let \mathbf{C} be a class of graphs.

- ▶ [Grohe, 2003] $\text{HOM}(\mathbf{C})$ is solvable in polynomial time if and only if *the cores of graphs in \mathbf{C} have bounded tree-width*.
- ▶ [Jonsson and Dalmau, 2004] $\#\text{HOM}(\mathbf{C})$ is solvable in polynomial time if and only if *the graphs in \mathbf{C} have bounded tree-width*.

What are known now

Assume $\mathbf{W[1]} \neq \mathbf{FPT}$:

Theorem

Let \mathbf{C} be a class of graphs.

- ▶ [Grohe, 2003] $\text{HOM}(\mathbf{C})$ is solvable in polynomial time if and only if *the cores of graphs in \mathbf{C} have bounded tree-width*.
- ▶ [Jonsson and Dalmau, 2004] $\#\text{HOM}(\mathbf{C})$ is solvable in polynomial time if and only if *the graphs in \mathbf{C} have bounded tree-width*.
- ▶ $\text{STREMB}(\mathbf{C})$ and $\#\text{STREMB}(\mathbf{C})$ are solvable in polynomial time if and only if *\mathbf{C} is finite*.

Subgraph Problems

Subgraph Problems

Definition

Let G, H be two graphs. A mapping $h : V(G) \rightarrow V(H)$ is an embedding if

Subgraph Problems

Definition

Let G, H be two graphs. A mapping $h : V(G) \rightarrow V(H)$ is an embedding if h is **injective**,

Subgraph Problems

Definition

Let G, H be two graphs. A mapping $h : V(G) \rightarrow V(H)$ is an embedding if h is **injective**, and for every $u, v, \in V(G)$ we have

$$\{u, v\} \in E(G) \implies \{h(u), h(v)\} \in E(H).$$

Subgraph Problems

Definition

Let G, H be two graphs. A mapping $h : V(G) \rightarrow V(H)$ is an embedding if h is **injective**, and for every $u, v, \in V(G)$ we have

$$\{u, v\} \in E(G) \implies \{h(u), h(v)\} \in E(H).$$

Definition

Let \mathbf{C} be a class of graphs. Then the parameterized embedding problem on \mathbf{C}

p -EMB(\mathbf{C})

Input: Two graphs G and H with $G \in \mathbf{C}$.

Parameter: $\|G\|$.

Problem: Is there an embedding from G to H ?

Subgraph Problems

Definition

Let G, H be two graphs. A mapping $h : V(G) \rightarrow V(H)$ is an embedding if h is **injective**, and for every $u, v, \in V(G)$ we have

$$\{u, v\} \in E(G) \implies \{h(u), h(v)\} \in E(H).$$

Definition

Let \mathbf{C} be a class of graphs. Then the parameterized embedding problem on \mathbf{C}

p -EMB(\mathbf{C})

Input: Two graphs G and H with $G \in \mathbf{C}$.

Parameter: $\|G\|$.

Problem: Is there an embedding from G to H ?

That is, H contains a **subgraph** isomorphic to G .

What are still open

What are still open

Conjecture *Let \mathbf{C} be a class of graphs.*

What are still open

Conjecture *Let \mathbf{C} be a class of graphs.*

- ▶ p -EMB(\mathbf{C}) is *fixed-parameter tractable* if and only if *the graphs in \mathbf{C} have bounded tree-width*.

What are still open

Conjecture *Let \mathbf{C} be a class of graphs.*

- ▶ p -EMB(\mathbf{C}) is *fixed-parameter tractable* if and only if *the graphs in \mathbf{C} have bounded tree-width*.

The converse direction is known [Alon, Yuster, and Zwick, 1995].

What are still open

Conjecture Let \mathbf{C} be a class of graphs.

- ▶ p -EMB(\mathbf{C}) is *fixed-parameter tractable* if and only if *the graphs in \mathbf{C} have bounded tree-width*.
The converse direction is known [Alon, Yuster, and Zwick, 1995].
- ▶ #EMB(\mathbf{C}) is solvable in polynomial time if and only if *the graphs in \mathbf{C} have bounded matching number*.

What are still open

Conjecture Let \mathbf{C} be a class of graphs.

- ▶ p -EMB(\mathbf{C}) is *fixed-parameter tractable* if and only if *the graphs in \mathbf{C} have bounded tree-width*.
The converse direction is known [Alon, Yuster, and Zwick, 1995].
- ▶ $\#$ EMB(\mathbf{C}) is solvable in polynomial time if and only if *the graphs in \mathbf{C} have bounded matching number*.
The converse direction is known [C. and Flum, 2007].

Thank You!