The first-order logic of graphs

Rose McCarty

Schools of Math and CS

Georgia Tech

June 13, 2025

Tutte Colloquium

Idea: We only care about graph properties that can be expressed in a certain language.

Idea: We only care about graph properties that can be expressed in a certain language.

In **first-order logic**, the language can:

- quantify over vertices ($\exists v \in V, \forall u \in V$),

Idea: We only care about graph properties that can be expressed in a certain language.

In **first-order logic**, the language can:

- quantify over vertices ($\exists v \in V, \forall u \in V$),
- use logical connectives ($\wedge, \vee, \neg, \implies, \ldots$), and

Idea: We only care about graph properties that can be expressed in a certain language.

In **first-order logic**, the language can:

- quantify over vertices ($\exists v \in V, \forall u \in V$),
- use logical connectives ($\wedge, \vee, \neg, \implies, \ldots$), and
- ask if two vertices are adjacent or equal (uEv, $u = v$).

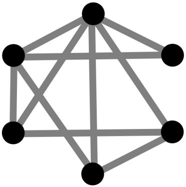
Idea: We only care about graph properties that can be expressed in a certain language.

In **first-order logic**, the language can:

- quantify over vertices ($\exists v \in V, \forall u \in V$),
- use logical connectives ($\wedge, \vee, \neg, \implies, \ldots$), and
- ask if two vertices are adjacent or equal (uEv, $u = v$).

The following **sentence** expresses "G has a universal vertex":

$$\phi = \exists v \in V : (\forall u \in V : (uEv \vee u = v)).$$



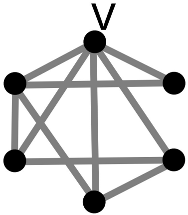
Idea: We only care about graph properties that can be expressed in a certain language.

In **first-order logic**, the language can:

- quantify over vertices ($\exists v \in V, \forall u \in V$),
- use logical connectives ($\wedge, \vee, \neg, \implies, \ldots$), and
- ask if two vertices are adjacent or equal (uEv, $u = v$).

The following **sentence** expresses "G has a universal vertex":

$$\phi = \exists v \in V : (\forall u \in V : (uEv \vee u = v)).$$



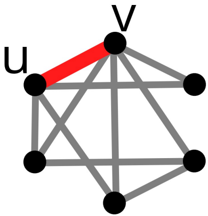
Idea: We only care about graph properties that can be expressed in a certain language.

In **first-order logic**, the language can:

- quantify over vertices ($\exists v \in V, \forall u \in V$),
- use logical connectives ($\wedge, \vee, \neg, \implies, \ldots$), and
- ask if two vertices are adjacent or equal (uEv, $u = v$).

The following **sentence** expresses "G has a universal vertex":

$$\phi = \exists v \in V : (\forall u \in V : (uEv \vee u = v)).$$



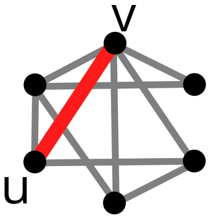
Idea: We only care about graph properties that can be expressed in a certain language.

In **first-order logic**, the language can:

- quantify over vertices ($\exists v \in V, \forall u \in V$),
- use logical connectives ($\wedge, \vee, \neg, \implies, \ldots$), and
- ask if two vertices are adjacent or equal ($uEv, u = v$).

The following **sentence** expresses "G has a universal vertex":

$$\phi = \exists v \in V : (\forall u \in V : (uEv \vee u = v)).$$



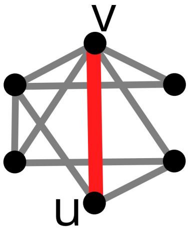
Idea: We only care about graph properties that can be expressed in a certain language.

In **first-order logic**, the language can:

- quantify over vertices ($\exists v \in V, \forall u \in V$),
- use logical connectives ($\wedge, \vee, \neg, \implies, \ldots$), and
- ask if two vertices are adjacent or equal ($uEv, u = v$).

The following **sentence** expresses "G has a universal vertex":

$$\phi = \exists v \in V : (\forall u \in V : (uEv \vee u = v)).$$



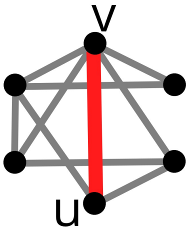
Idea: We only care about graph properties that can be expressed in a certain language.

In **first-order logic**, the language can:

- quantify over vertices ($\exists v \in V, \forall u \in V$),
- use logical connectives ($\wedge, \vee, \neg, \implies, \ldots$), and
- ask if two vertices are adjacent or equal (uEv, $u = v$).

The following **sentence** expresses "G has a universal vertex":

$$\phi = \exists v \in V : (\forall u \in V : (uEv \vee u = v)).$$



Idea: We only care about graph properties that can be expressed in a certain language.

In **first-order logic**, the language can:

- quantify over vertices ($\exists v \in V, \forall u \in V$),
- use logical connectives ($\wedge, \vee, \neg, \implies, \ldots$), and
- ask if two vertices are adjacent or equal (uEv, $u = v$).

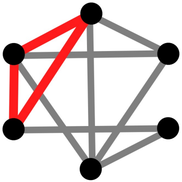
The following **sentence** expresses "G has a universal vertex":

$$\phi = \exists v \in V : (\forall u \in V : (uEv \vee u = v)).$$

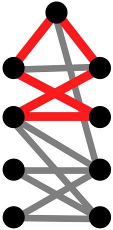
If ϕ is true for G, we write $G \models \phi$ and say G **models** ϕ.

Graph Properties

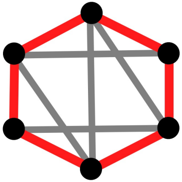
contains triangle



contains odd cycle



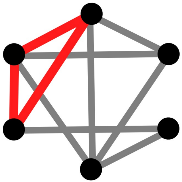
contains Hamiltonian cycle



$$G = (V, E)$$

Graph Properties

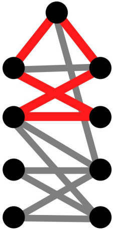
contains triangle



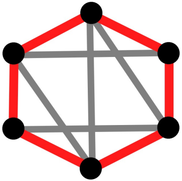
$\exists a, b, c \in V \ldots$

first-order

contains odd cycle



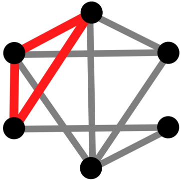
contains Hamiltonian cycle



$$G = (V, E)$$

Graph Properties

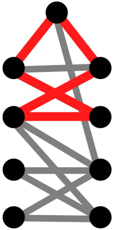
contains triangle



$\exists a, b, c \in V \ldots$

first-order

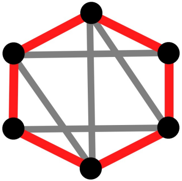
contains odd cycle



$\nexists X, Y \subseteq V \ldots$

$\mathbf{MSO_1}$

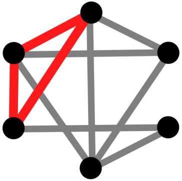
contains Hamiltonian cycle



$$G = (V, E)$$

Graph Properties

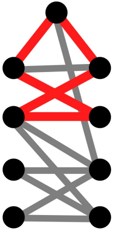
contains triangle



$\exists a, b, c \in V \dots$

first-order

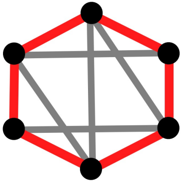
contains odd cycle



$\nexists X, Y \subseteq V \dots$

$\mathbf{MSO_1}$

contains Hamiltonian cycle



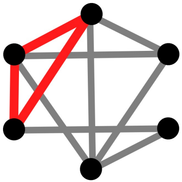
$\exists F \subseteq E \dots$

$\mathbf{MSO_2}$

$$G = (V, E)$$

Graph Properties

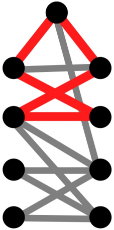
contains
triangle



$\exists a, b, c \in V \dots$

first-order

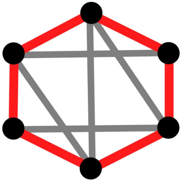
contains
odd cycle



$\nexists X, Y \subseteq V \dots$

$\mathbf{MSO_1}$

contains
Hamiltonian
cycle



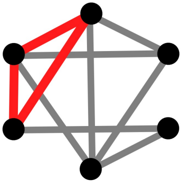
$\exists F \subseteq E \dots$

$\mathbf{MSO_2}$

Lemma: There is no $\mathbf{MSO_1}$ sentence ϕ_{Ham} which expresses "G contains a Hamiltonian cycle".

Graph Properties

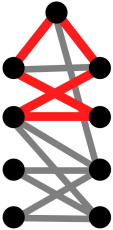
contains triangle



$\exists a, b, c \in V \dots$

first-order

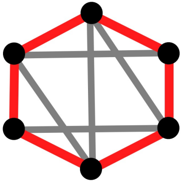
contains odd cycle



$\nexists X, Y \subseteq V \dots$

MSO$_1$

contains Hamiltonian cycle



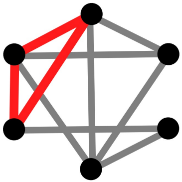
$\exists F \subseteq E \dots$

MSO$_2$

Lemma: There is no **first-order** sentence ϕ_{odd} which expresses "G contains an odd cycle".

Graph Properties

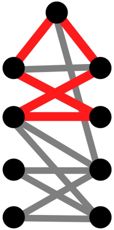
contains triangle



$\exists a, b, c \in V \ldots$

first-order

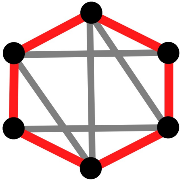
contains odd cycle



$\nexists X, Y \subseteq V \ldots$

$\mathbf{MSO_1}$

contains Hamiltonian cycle



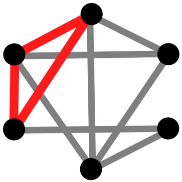
$\exists F \subseteq E \ldots$

$\mathbf{MSO_2}$

expressive power →

Graph Properties

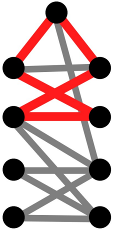
contains triangle



$\exists a, b, c \in V \dots$

first-order

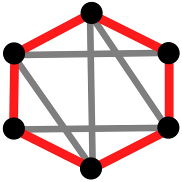
contains odd cycle



$\nexists X, Y \subseteq V \dots$

MSO_1

contains Hamiltonian cycle



$\exists F \subseteq E \dots$

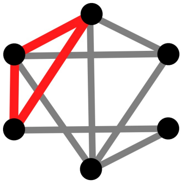
MSO_2

expressive power →

← computational tractability

Graph Properties

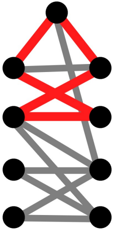
contains triangle



$\exists a, b, c \in V \ldots$

first-order

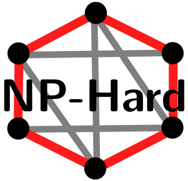
contains odd cycle



$\nexists X, Y \subseteq V \ldots$

MSO$_1$

contains Hamiltonian cycle



$\exists F \subseteq E \ldots$

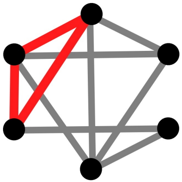
MSO$_2$

expressive power →

← computational tractability

Graph Properties

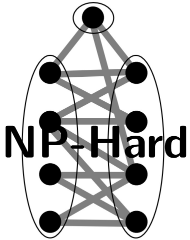
contains triangle



$\exists a, b, c \in V \ldots$

first-order

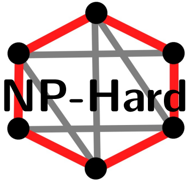
is 3-colorable



$\exists X, Y, Z \subseteq V \ldots$

MSO_1

contains Hamiltonian cycle



$\exists F \subseteq E \ldots$

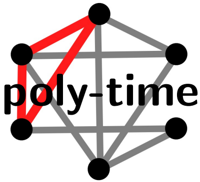
MSO_2

expressive power →

← computational tractability

Graph Properties

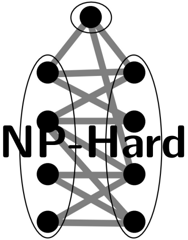
contains triangle



$\exists a, b, c \in V \ldots$

first-order

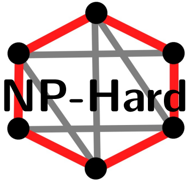
is 3-colorable



$\exists X, Y, Z \subseteq V \ldots$

MSO$_1$

contains Hamiltonian cycle



$\exists F \subseteq E \ldots$

MSO$_2$

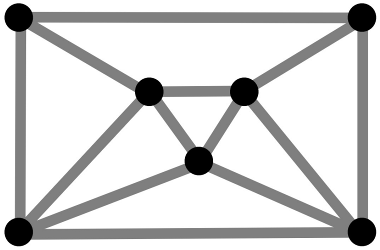
expressive power →

← computational tractability

Does $G \models \phi$?

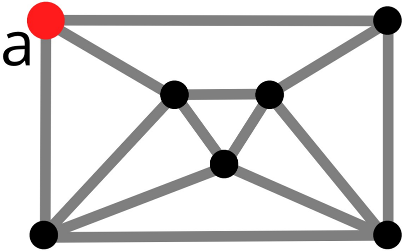
Does $G \models \phi$?

Consider $\phi = \exists a \exists b \exists c (aEb \wedge aEc \wedge bEc)$.



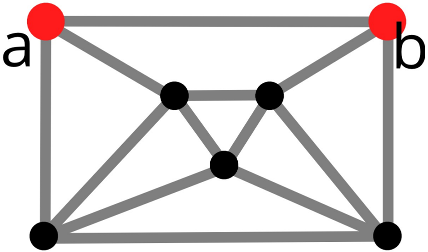
Does $G \models \phi$?

Consider $\phi = \exists a \exists b \exists c (aEb \wedge aEc \wedge bEc)$.



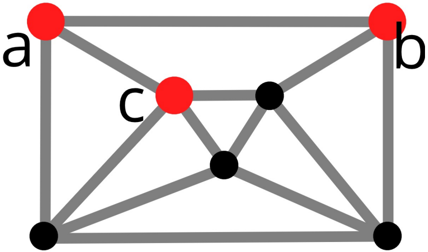
Does $G \models \phi$?

Consider $\phi = \exists a \exists b \exists c (aEb \wedge aEc \wedge bEc)$.



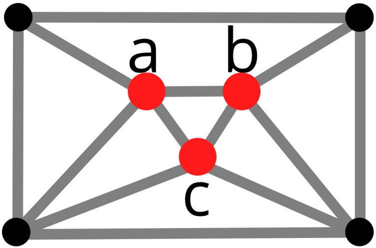
Does $G \models \phi$?

Consider $\phi = \exists a \exists b \exists c (aEb \wedge aEc \wedge bEc)$.



Does $G \models \phi$?

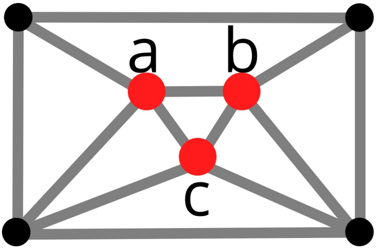
Consider $\phi = \exists a \exists b \exists c (aEb \wedge aEc \wedge bEc)$.



Does $G \models \phi$?

Consider $\phi = \exists a \exists b \exists c (aEb \wedge aEc \wedge bEc)$.

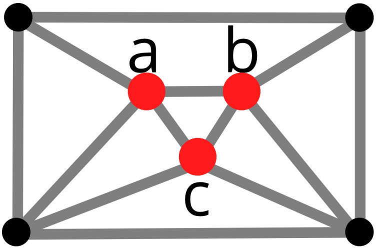
Naive takes time $\mathcal{O}(n^3)$.



Does $G \models \phi$?

Consider $\phi = \exists a \exists b \exists c (aEb \wedge aEc \wedge bEc)$.

Naive takes time $\mathcal{O}(n^3)$.

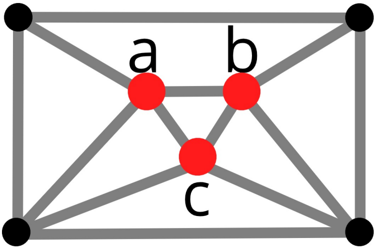


Matrix multiplication gives $\mathcal{O}(n^{2.372})$.

Does $G \models \phi$?

Consider $\phi = \exists a \exists b \exists c (aEb \wedge aEc \wedge bEc)$.

Naive takes time $\mathcal{O}(n^3)$.



Matrix multiplication gives $\mathcal{O}(n^{2.372})$.

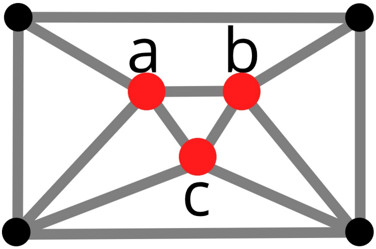
Theorem (Dvořák, Král', Thomas 2011)

*Can check for triangles in time $\mathcal{O}(n)$ if G is **planar**.*

Does $G \models \phi$?

Consider $\phi = \exists a \exists b \exists c (aEb \wedge aEc \wedge bEc)$.

Naive takes time $\mathcal{O}(n^3)$.



Matrix multiplication gives $\mathcal{O}(n^{2.372})$.

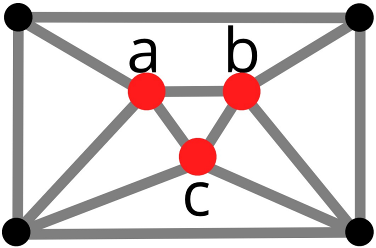
Theorem (Dvořák, Král', Thomas 2011)

For each **first-order** *ϕ, can do in time $\mathcal{O}_\phi(n)$ if G is* **planar**.

Does $G \models \phi$?

Consider $\phi = \exists a \exists b \exists c (aEb \wedge aEc \wedge bEc)$.

Naive takes time $\mathcal{O}(n^3)$.



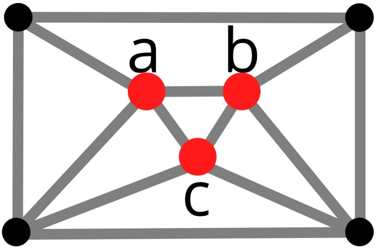
Matrix multiplication gives $\mathcal{O}(n^{2.372})$.

Efficient: $\exists d$ s.t. we can check whether an n-vertex $G \in \mathcal{F}$ models an FO sentence ϕ in time $\mathcal{O}_\phi(n^d)$.

Does $G \models \phi$?

Consider $\phi = \exists a \exists b \exists c (aEb \wedge aEc \wedge bEc)$.

Naive takes time $\mathcal{O}(n^3)$.

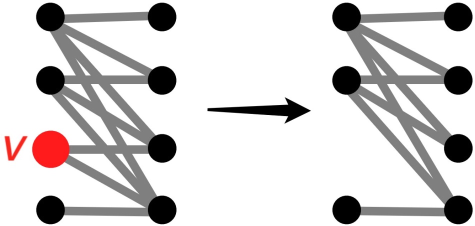


Matrix multiplication gives $\mathcal{O}(n^{2.372})$.

Efficient: $\exists d$ s.t. we can check whether an n-vertex $G \in \mathcal{F}$ models an FO sentence ϕ in time $\mathcal{O}_\phi(n^d)$.

i.e. $\mathcal{O}_\phi(n^6)$

From now on we assume $\mathcal{F}$ is closed under vertex-deletion.

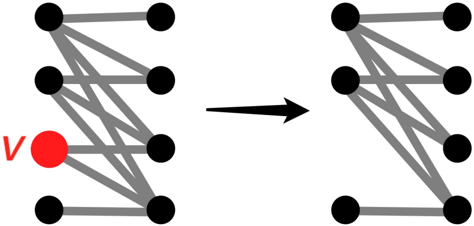


$G \in \mathcal{F}$ $\qquad$ $G - v \in \mathcal{F}$

From now on we assume $\mathcal{F}$ is closed under vertex-deletion.

Conjecture (folklore)

A class $\mathcal{F}$ admits an **efficient algorithm** *if and only if it has good* **model-theoretic** *properties in the sense of Shelah.*



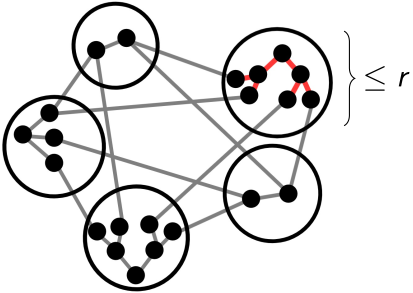
$G \in \mathcal{F}$ $\qquad$ $G - v \in \mathcal{F}$

From now on we assume $\mathcal{F}$ is closed under vertex-deletion.

Conjecture (folklore)

A class $\mathcal{F}$ admits an **efficient algorithm** *if and only if it has good* **model-theoretic** *properties in the sense of Shelah.*

1) True if the class is **sparse** (i.e. has $o(n^2)$ edges).



From now on we assume $\mathcal{F}$ is closed under vertex-deletion.

Conjecture (folklore)

A class $\mathcal{F}$ admits an **efficient algorithm** *if and only if it has good* **model-theoretic** *properties in the sense of Shelah.*

1) True if the class is **sparse** (i.e. has $o(n^2)$ edges).
    a) (Dvořák, Král' & Thomas 2011) proved hardness.

From now on we assume $\mathcal{F}$ is closed under vertex-deletion.

Conjecture (folklore)

A class $\mathcal{F}$ admits an **efficient algorithm** *if and only if it has good* **model-theoretic** *properties in the sense of Shelah.*

1) True if the class is **sparse** (i.e. has $o(n^2)$ edges).
   a) (Dvořák, Král' & Thomas 2011) proved hardness.
   b) (Adler & Adler 2014) found the connection to model theory.

From now on we assume $\mathcal{F}$ is closed under vertex-deletion.

Conjecture (folklore)

A class $\mathcal{F}$ admits an **efficient algorithm** *if and only if it has good* **model-theoretic** *properties in the sense of Shelah.*

1) True if the class is **sparse** (i.e. has $o(n^2)$ edges).
   a) (Dvořák, Král' & Thomas 2011) proved hardness.
   b) (Adler & Adler 2014) found the connection to model theory.
   c) (Grohe, Kreutzer & Siebertz 2017) found the algorithm.

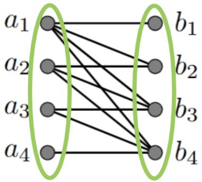
From now on we assume $\mathcal{F}$ is closed under vertex-deletion.

Conjecture (folklore)

A class $\mathcal{F}$ admits an **efficient algorithm** *if and only if it has good* **model-theoretic** *properties in the sense of Shelah.*

1) True if the class is **sparse** (i.e. has $o(n^2)$ edges).
   a) (Dvořák, Král' & Thomas 2011) proved hardness.
   b) (Adler & Adler 2014) found the connection to model theory.
   c) (Grohe, Kreutzer & Siebertz 2017) found the algorithm.

2) True if the class is **edge-stable**.



From now on we assume $\mathcal{F}$ is closed under vertex-deletion.

Conjecture (folklore)

A class $\mathcal{F}$ admits an **efficient algorithm** *if and only if it has good* **model-theoretic** *properties in the sense of Shelah.*

1) True if the class is **sparse** (i.e. has $o(n^2)$ edges).
   a) (Dvořák, Král' & Thomas 2011) proved hardness.
   b) (Adler & Adler 2014) found the connection to model theory.
   c) (Grohe, Kreutzer & Siebertz 2017) found the algorithm.
2) True if the class is **edge-stable**.
   a) (Gajarský, Mählmann, McCarty, Ohlmann, Pilipczuk, Przybyszewski, Siebertz, Sokołowski & Toruńczyk 2023) proved an equivalent characterization via a **combinatorial game**.

From now on we assume $\mathcal{F}$ is closed under vertex-deletion.

Conjecture (folklore)

A class $\mathcal{F}$ admits an **efficient algorithm** *if and only if it has good* **model-theoretic** *properties in the sense of Shelah.*

1) True if the class is **sparse** (i.e. has $o(n^2)$ edges).
   a) (Dvořák, Král' & Thomas 2011) proved hardness.
   b) (Adler & Adler 2014) found the connection to model theory.
   c) (Grohe, Kreutzer & Siebertz 2017) found the algorithm.
2) True if the class is **edge-stable**.
   a) (Gajarský, Mählmann, McCarty, Ohlmann, Pilipczuk, Przybyszewski, Siebertz, Sokołowski & Toruńczyk 2023) proved an equivalent characterization via a **combinatorial game**.
   b) (Dreier, Mählmann & Siebertz 2023) found the algorithm that *should* work.

From now on we assume $\mathcal{F}$ is closed under vertex-deletion.

Conjecture (folklore)

*A class $\mathcal{F}$ admits an **efficient algorithm** if and only if it has good **model-theoretic** properties in the sense of Shelah.*

1) True if the class is **sparse** (i.e. has $o(n^2)$ edges).
   a) (Dvořák, Král' & Thomas 2011) proved hardness.
   b) (Adler & Adler 2014) found the connection to model theory.
   c) (Grohe, Kreutzer & Siebertz 2017) found the algorithm.
2) True if the class is **edge-stable**.
   a) (Gajarský, Mählmann, McCarty, Ohlmann, Pilipczuk, Przybyszewski, Siebertz, Sokołowski & Toruńczyk 2023) proved an equivalent characterization via a **combinatorial game**.
   b) (Dreier, Mählmann & Siebertz 2023) found the algorithm.
   c) (Dreier, Eleftheriadis, Mählmann, McCarty, Pilipczuk & Toruńczyk 2024) proved hardness and the missing link.

Next:

1) What are “good **model-theoretic** properties”?

Next:

1) What are "good **model-theoretic** properties"?
2) The key ingredients to the algorithm.

Next:

1) What are “good **model-theoretic** properties”?
2) The key ingredients to the algorithm.
    a) The **flipper game**.

Theorem (GMMOPPSST 2023)

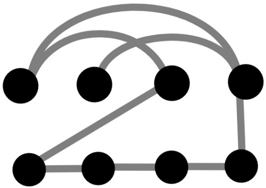
A class of graphs is **stable** *if and only if Flipper wins the* **radius-r flipper game** *for each $r \in \mathbb{N}$.*

Two player game: Flipper and Connector.

Theorem (GMMOPPSST 2023)

A class of graphs is **stable** *if and only if Flipper wins the* **radius-r flipper game** *for each $r \in \mathbb{N}$.*

Two player game: Flipper and Connector.

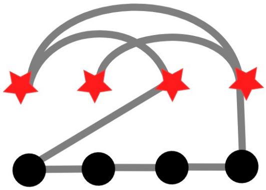


To **flip,** select $X \subseteq V(G)$ and replace $G[X]$ by its complement.

Theorem (GMMOPPSST 2023)

A class of graphs is **stable** *if and only if Flipper wins the* **radius-r flipper game** *for each $r \in \mathbb{N}$.*

Two player game: Flipper and Connector.

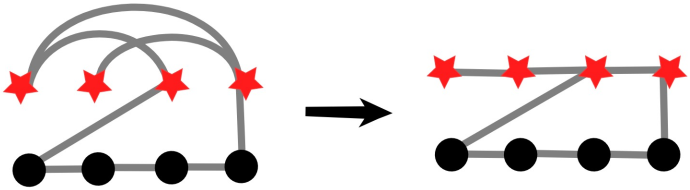


To **flip,** select $X \subseteq V(G)$ and replace $G[X]$ by its complement.

Theorem (GMMOPPSST 2023)

A class of graphs is **stable** *if and only if Flipper wins the* **radius-r flipper game** *for each $r \in \mathbb{N}$.*

Two player game: Flipper and Connector.



To **flip,** select $X \subseteq V(G)$ and replace $G[X]$ by its complement.

Theorem (GMMOPPSST 2023)

A class of graphs is **stable** *if and only if Flipper wins the* **radius-r flipper game** *for each $r \in \mathbb{N}$.*

Two player game: Flipper and Connector. In each round:

1) If $|V(G)| = 1$ then Flipper wins in that round.

Theorem (GMMOPPSST 2023)

A class of graphs is **stable** *if and only if Flipper wins the* **radius-r flipper game** *for each $r \in \mathbb{N}$.*

Two player game: Flipper and Connector. In each round:

1) If $|V(G)| = 1$ then Flipper wins in that round.
2) Else, Connector picks a vertex v and we restrict to $B_r(v)$.

Theorem (GMMOPPSST 2023)

A class of graphs is **stable** *if and only if Flipper wins the* **radius-r flipper game** *for each $r \in \mathbb{N}$.*

Two player game: Flipper and Connector. In each round:

1) If $|V(G)| = 1$ then Flipper wins in that round.
2) Else, Connector picks a vertex v and we restrict to $B_r(v)$.
3) Then Flipper performs a flip.

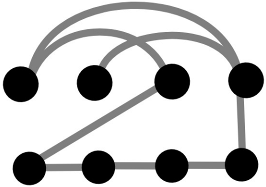
Theorem (GMMOPPSST 2023)

A class of graphs is **stable** *if and only if Flipper wins the* **radius-r flipper game** *for each $r \in \mathbb{N}$.*

Two player game: Flipper and Connector. In each round:

1) If $|V(G)| = 1$ then Flipper wins in that round.
2) Else, Connector picks a vertex v and we restrict to $B_r(v)$.
3) Then Flipper performs a flip.

Example with $r = 2$:



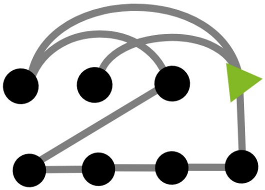
Theorem (GMMOPPSST 2023)

A class of graphs is **stable** *if and only if Flipper wins the* **radius-r flipper game** *for each $r \in \mathbb{N}$.*

Two player game: Flipper and Connector. In each round:

1) If $|V(G)| = 1$ then Flipper wins in that round.
2) Else, Connector picks a vertex v and we restrict to $B_r(v)$.
3) Then Flipper performs a flip.

Example with $r = 2$:



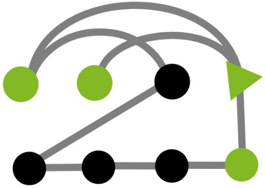
Theorem (GMMOPPSST 2023)

A class of graphs is **stable** *if and only if Flipper wins the* **radius-r flipper game** *for each $r \in \mathbb{N}$.*

Two player game: Flipper and Connector. In each round:

1) If $|V(G)| = 1$ then Flipper wins in that round.
2) Else, Connector picks a vertex v and we restrict to $B_r(v)$.
3) Then Flipper performs a flip.

Example with $r = 2$:



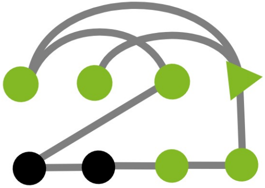
Theorem (GMMOPPSST 2023)

A class of graphs is **stable** *if and only if Flipper wins the* **radius-r flipper game** *for each $r \in \mathbb{N}$.*

Two player game: Flipper and Connector. In each round:

1) If $|V(G)| = 1$ then Flipper wins in that round.
2) Else, Connector picks a vertex v and we restrict to $B_r(v)$.
3) Then Flipper performs a flip.

Example with $r = 2$:



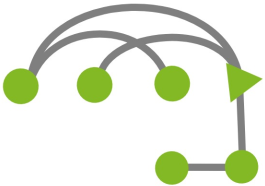
Theorem (GMMOPPSST 2023)

A class of graphs is **stable** *if and only if Flipper wins the* **radius-r flipper game** *for each $r \in \mathbb{N}$.*

Two player game: Flipper and Connector. In each round:

1) If $|V(G)| = 1$ then Flipper wins in that round.
2) Else, Connector picks a vertex v and we restrict to $B_r(v)$.
3) Then Flipper performs a flip.

Example with $r = 2$:



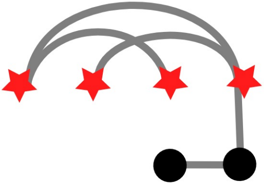
Theorem (GMMOPPSST 2023)

A class of graphs is **stable** *if and only if Flipper wins the* **radius-r flipper game** *for each $r \in \mathbb{N}$.*

Two player game: Flipper and Connector. In each round:

1) If $|V(G)| = 1$ then Flipper wins in that round.
2) Else, Connector picks a vertex v and we restrict to $B_r(v)$.
3) Then Flipper performs a flip.

Example with $r = 2$:



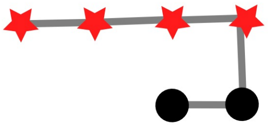
Theorem (GMMOPPSST 2023)

A class of graphs is **stable** *if and only if Flipper wins the* **radius-r flipper game** *for each $r \in \mathbb{N}$.*

Two player game: Flipper and Connector. In each round:

1) If $|V(G)| = 1$ then Flipper wins in that round.
2) Else, Connector picks a vertex v and we restrict to $B_r(v)$.
3) Then Flipper performs a flip.

Example with $r = 2$:



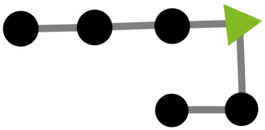
Theorem (GMMOPPSST 2023)

A class of graphs is **stable** *if and only if Flipper wins the* **radius-r flipper game** *for each $r \in \mathbb{N}$.*

Two player game: Flipper and Connector. In each round:

1) If $|V(G)| = 1$ then Flipper wins in that round.
2) Else, Connector picks a vertex v and we restrict to $B_r(v)$.
3) Then Flipper performs a flip.

Example with $r = 2$:



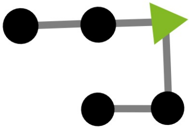
Theorem (GMMOPPSST 2023)

A class of graphs is **stable** *if and only if Flipper wins the* **radius-r flipper game** *for each $r \in \mathbb{N}$.*

Two player game: Flipper and Connector. In each round:

1) If $|V(G)| = 1$ then Flipper wins in that round.
2) Else, Connector picks a vertex v and we restrict to $B_r(v)$.
3) Then Flipper performs a flip.

Example with $r = 2$:



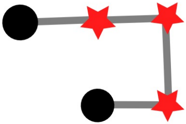
Theorem (GMMOPPSST 2023)

A class of graphs is **stable** *if and only if Flipper wins the* **radius-r flipper game** *for each $r \in \mathbb{N}$.*

Two player game: Flipper and Connector. In each round:

1) If $|V(G)| = 1$ then Flipper wins in that round.
2) Else, Connector picks a vertex v and we restrict to $B_r(v)$.
3) Then Flipper performs a flip.

Example with $r = 2$:



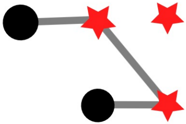
Theorem (GMMOPPSST 2023)

A class of graphs is **stable** *if and only if Flipper wins the* **radius-r flipper game** *for each $r \in \mathbb{N}$.*

Two player game: Flipper and Connector. In each round:

1) If $|V(G)| = 1$ then Flipper wins in that round.
2) Else, Connector picks a vertex v and we restrict to $B_r(v)$.
3) Then Flipper performs a flip.

Example with $r = 2$:



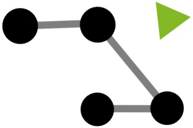
Theorem (GMMOPPSST 2023)

A class of graphs is **stable** *if and only if Flipper wins the* **radius-r flipper game** *for each $r \in \mathbb{N}$.*

Two player game: Flipper and Connector. In each round:

1) If $|V(G)| = 1$ then Flipper wins in that round.
2) Else, Connector picks a vertex v and we restrict to $B_r(v)$.
3) Then Flipper performs a flip.

Example with $r = 2$:



Theorem (GMMOPPSST 2023)

A class of graphs is **stable** *if and only if Flipper wins the* **radius-r flipper game** *for each $r \in \mathbb{N}$.*

Two player game: Flipper and Connector. In each round:

1) If $|V(G)| = 1$ then Flipper wins in that round.
2) Else, Connector picks a vertex v and we restrict to $B_r(v)$.
3) Then Flipper performs a flip.

Example with $r = 2$:

Theorem (GMMOPPSST 2023)

A class of graphs is **stable** *if and only if Flipper wins the* **radius-r flipper game** *for each $r \in \mathbb{N}$.*

Two player game: Flipper and Connector. In each round:

1) If $|V(G)| = 1$ then Flipper wins in that round.
2) Else, Connector picks a vertex v and we restrict to $B_r(v)$.
3) Then Flipper performs a flip.

Flipper **wins the game** on a class $\mathcal{F}$ if there exists $t = t(r)$ so that Flipper wins in $\leq t$ rounds on each $G \in \mathcal{F}$.

Next:

1) What are "good **model-theoretic** properties"?
2) The key ingredients to the algorithm.
   a) The **flipper game**.

Next:

1) What are "good **model-theoretic** properties"?
2) The key ingredients to the algorithm.
   a) The **flipper game**.
   b) Locality, pruning, and LP rounding.

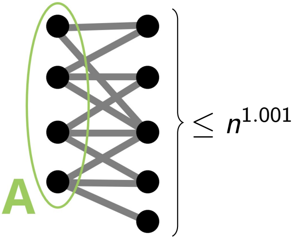
Next:

1) What are "good **model-theoretic** properties"?
2) The key ingredients to the algorithm.
   a) The **flipper game**.
   b) Locality, pruning, and LP rounding.
   c) Small **neighborhood complexity**.

Theorem (DEMMPT 2024)

If $\mathcal{F}$ is **stable**, *then for any $G \in \mathcal{F}$ and any set* **A** *of n vertices,*

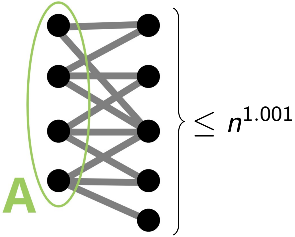
$$|\{N(v) \cap \mathbf{A} : v \in V(G)\}| \leq n^{1.001}.$$



Theorem (DEMMPT 2024)

If $\mathcal{F}$ is **stable**, *then for any $G \in \mathcal{F}$ and any set* **A** *of n vertices,*

$$|\{N(v) \cap \mathbf{A} : v \in V(G)\}| \leq n^{1.001}.$$



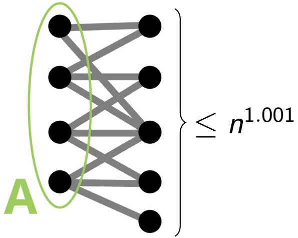
Conjecture (folklore)

The same bound holds if $\mathcal{F}$ is **dependent**.

Theorem (DEMMPT 2024)

If $\mathcal{F}$ is **stable**, *then for any $G \in \mathcal{F}$ and any set* **A** *of n vertices,*

$$|\{N(v) \cap \mathbf{A} : v \in V(G)\}| \leq n^{1.001}.$$



Conjecture (folklore)

The same bound holds if $\mathcal{F}$ is **dependent**.

Conjecture (Ossona de Mendez 2021)

Every **stable** *class is interdefinable with a* **sparse** *class.*

Thank you!