Double-exponential lower bounds for graph identification problems

Florent Foucaud
joint works with:

Esther Galby, Liana Khazaliya, Shaohua Li, Fionn Mc Inerney, Roohani Sharma, Prafullkumar Tale (ICALP 2024)

Dipayan Chakraborty, Diptapriyo Majumdar, Prafullkumar Tale (ISAAC 2024)

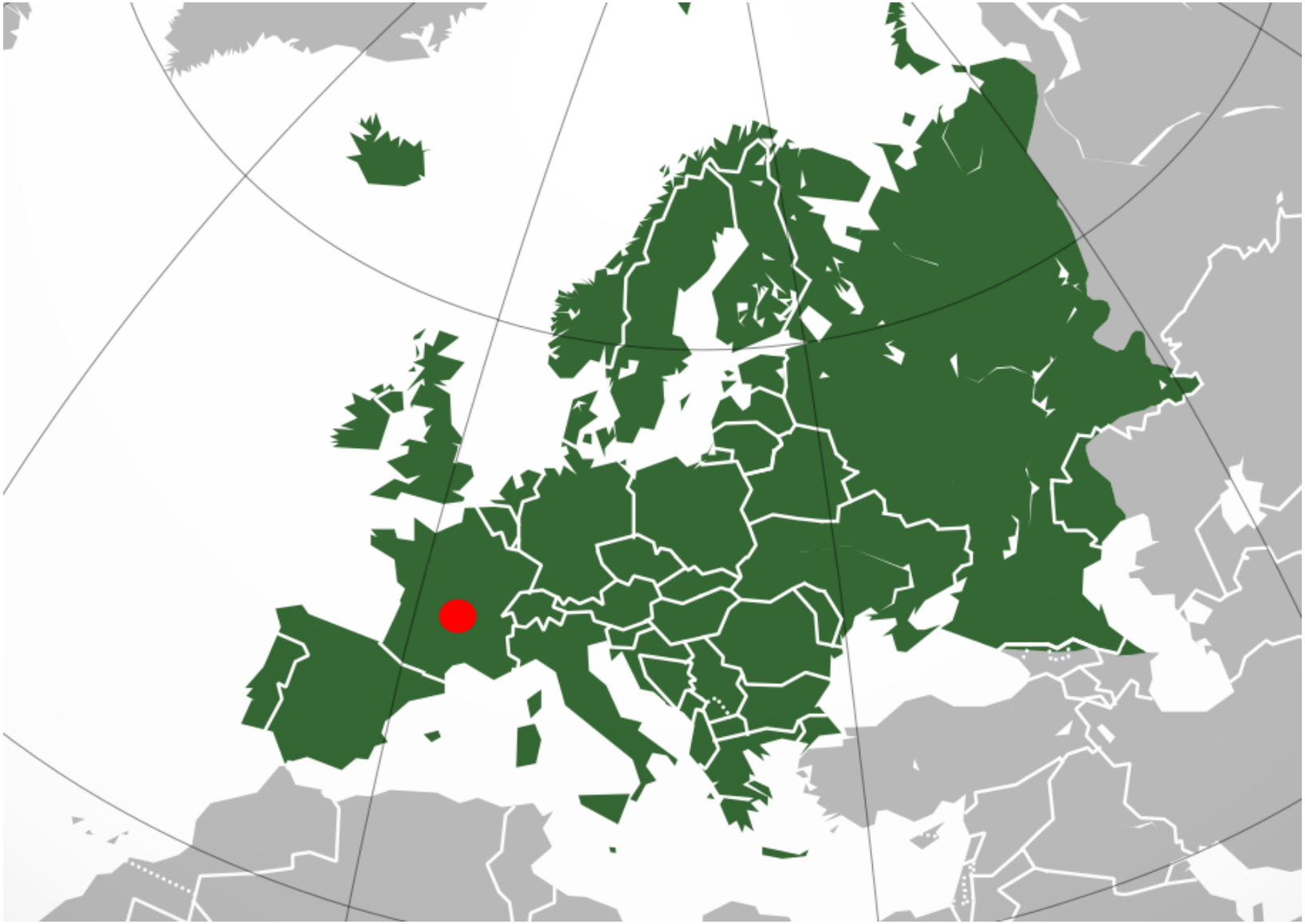
UNIVERSITÉ
Clermont Auvergne

January 2026

IWOCA 2026 in Clermont-Ferrand

- 37th International Workshop on Combinatorial Algorithms.
- Yearly since 1989, started in Australia.
- Submission deadline: January 26 (abstract), February 2 (full paper)
- Conference dates: June 8-12, 2026
- Collaborative aspect
- Website: `https://iwoca2026.limos.fr`

Where: Clermont-Ferrand, France



Where: Clermont-Ferrand, France

Where: Clermont-Ferrand, France


MICHELIN

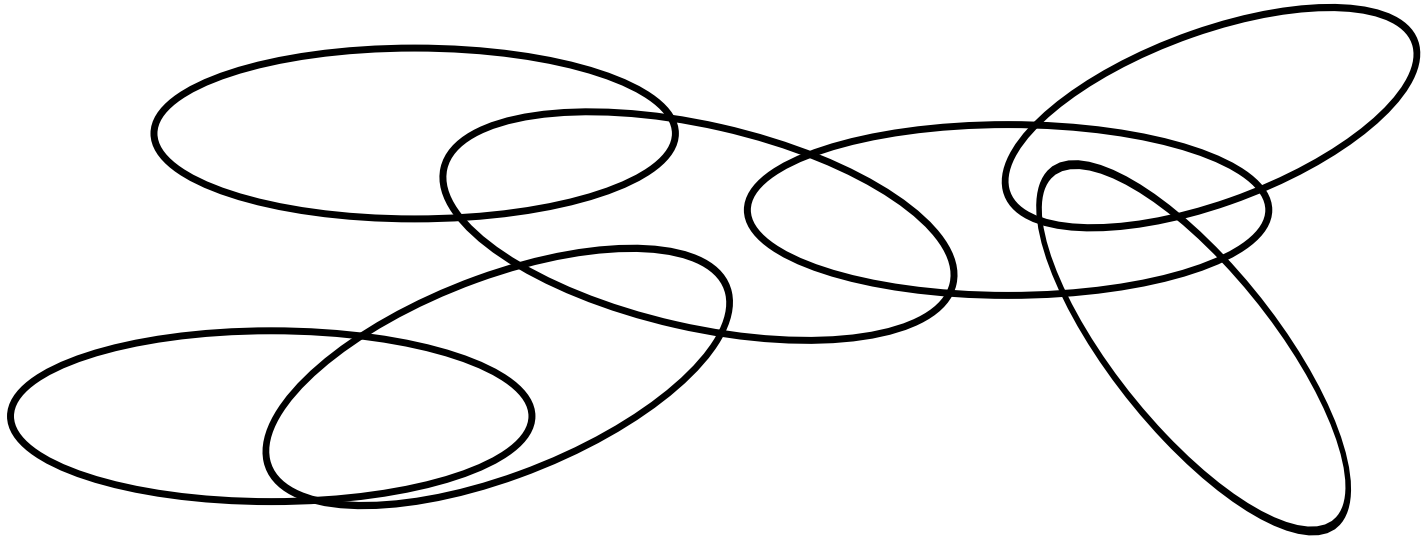

Saint
Nectaire

(START OF THE TALK)

Treewidth

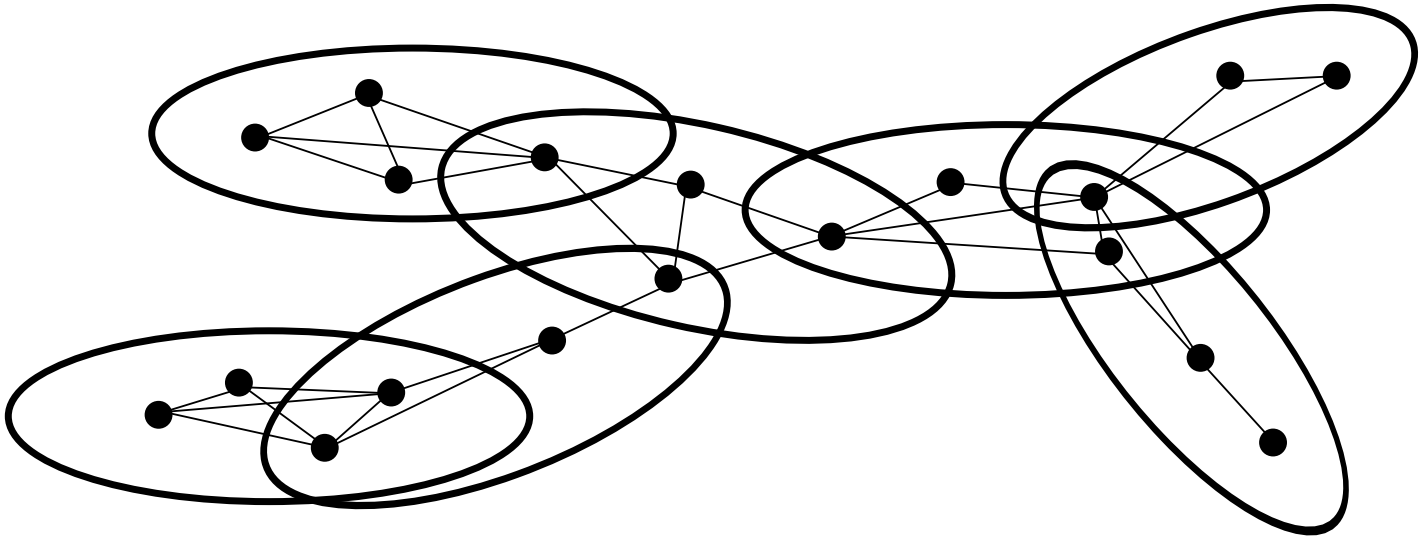
Treewidth

Graph G of treewidth k: looks like a tree where edges are replaced by "bags" of size k.



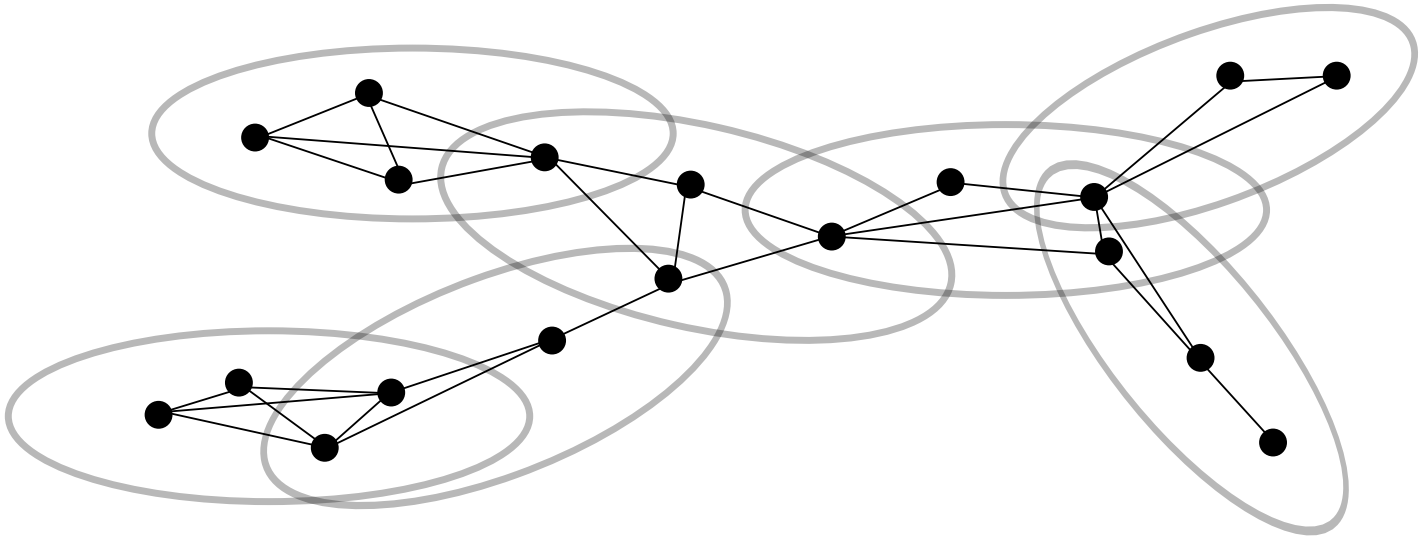
Treewidth

Graph G of treewidth k: looks like a tree where edges are replaced by "bags" of size k.



Treewidth

Graph G of treewidth k: looks like a tree where edges are replaced by "bags" of size k.

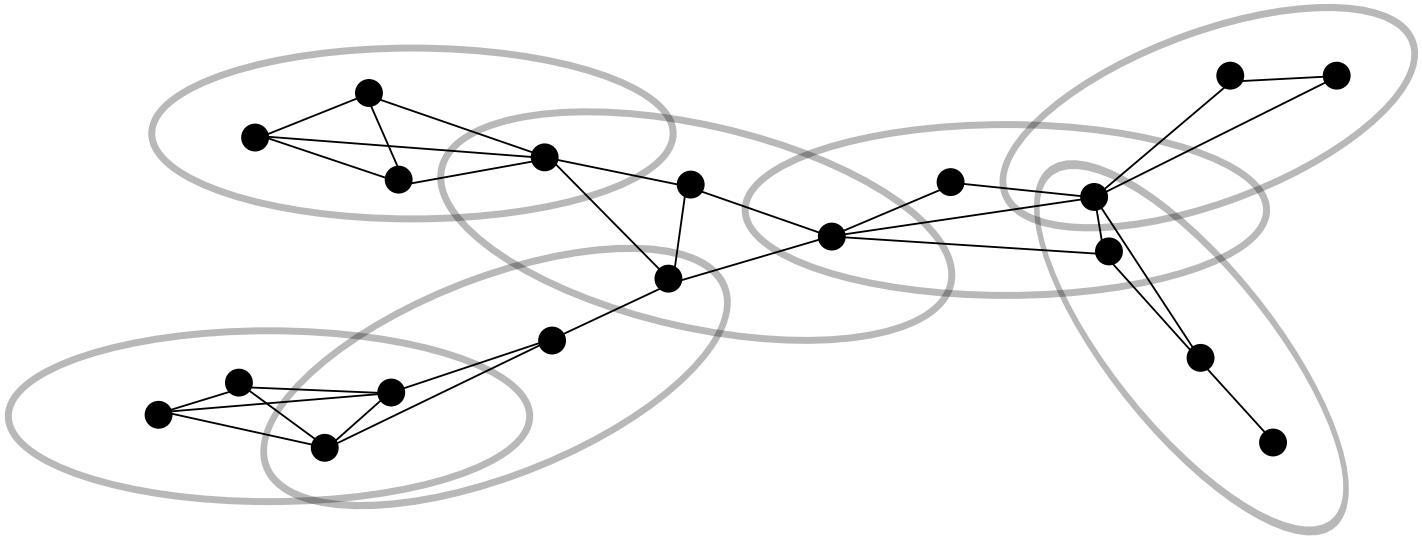


Informally:

- this tree forms a **tree-decomposition** of G
- its width is the largest size of a bag
- **treewidth** $\mathtt{tw}(G)$ of G is the smallest width over all tree-decompositions

Treewidth

Graph G of treewidth k: looks like a tree where edges are replaced by "bags" of size k.



Informally:

- this tree forms a **tree-decomposition** of G
- its width is the largest size of a bag
- **treewidth** $\mathtt{tw}(G)$ of G is the smallest width over all tree-decompositions

Very important in:

- Structural graph theory
- Graph algorithms
- Many applications in other areas: Database queries, Optimization, Boolean SAT...

Treewidth: the **King** of Structural Parameters

Fixed parameter tractable (FPT) problems

A **decision problem** with input $\mathcal{I}$ and **parameter** k is FPT parameterized by k if it can be solved in time $f(k) \cdot |\mathcal{I}|^{O(1)}$, where f is a computable function.

Treewidth: the **King** of Structural Parameters

Fixed parameter tractable (FPT) problems

A **decision problem** with input $\mathcal{I}$ and **parameter** k is FPT parameterized by k if it can be solved in time $f(k) \cdot |\mathcal{I}|^{O(1)}$, where f is a computable function.

Many NP-hard problems are FPT parameterized by treewidth via **dynamic programming** on a **tree-decomposition**: running time $f(\mathtt{tw}) \cdot n^{O(1)}$.

Treewidth: the **King** of Structural Parameters

Fixed parameter tractable (FPT) problems

A **decision problem** with input $\mathcal{I}$ and **parameter** k is FPT parameterized by k if it can be solved in time $f(k) \cdot |\mathcal{I}|^{O(1)}$, where f is a computable function.

Many NP-hard problems are FPT parameterized by treewidth via **dynamic programming** on a **tree-decomposition**: running time $f(\mathtt{tw}) \cdot n^{O(1)}$.

Classic examples: Colouring, Clique, Dominating Set, Hamilton Cycle...

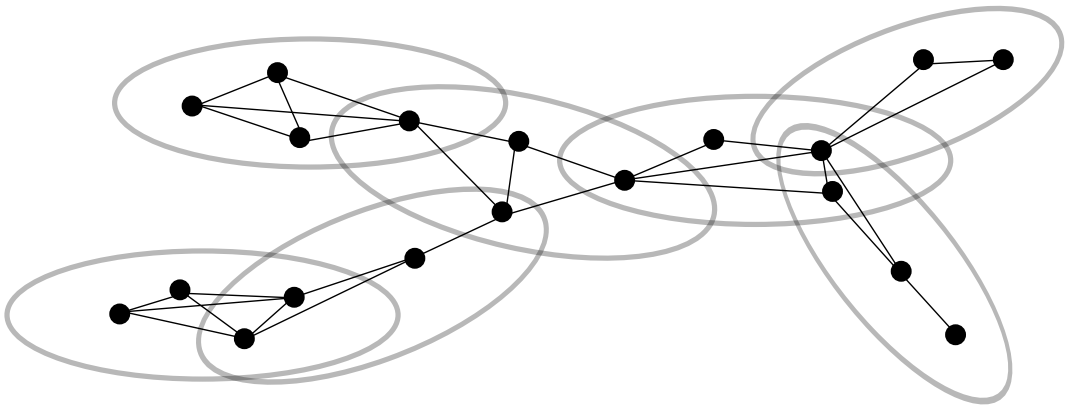
Treewidth: the **King** of Structural Parameters

Fixed parameter tractable (FPT) problems

A **decision problem** with input $\mathcal{I}$ and **parameter** k is FPT parameterized by k if it can be solved in time $f(k) \cdot |\mathcal{I}|^{O(1)}$, where f is a computable function.

Many NP-hard problems are FPT parameterized by treewidth via **dynamic programming** on a **tree-decomposition**: running time $f(\mathtt{tw}) \cdot n^{O(1)}$.

Classic examples: Colouring, Clique, Dominating Set, Hamilton Cycle...



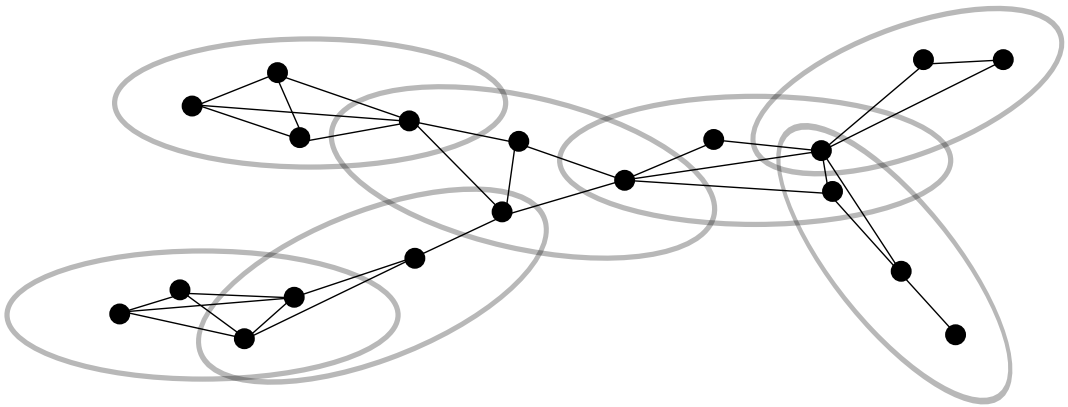
Treewidth: the **King** of Structural Parameters

Fixed parameter tractable (FPT) problems

A **decision problem** with input $\mathcal{I}$ and **parameter** k is FPT parameterized by k if it can be solved in time $f(k) \cdot |\mathcal{I}|^{O(1)}$, where f is a computable function.

Many NP-hard problems are FPT parameterized by treewidth via **dynamic programming** on a **tree-decomposition**: running time $f(\mathtt{tw}) \cdot n^{O(1)}$.

Classic examples: Colouring, Clique, Dominating Set, Hamilton Cycle...



Generally: graph problems expressible in **Monadic Second-Order (MSO) logic** are FPT parameterized by the treewidth plus the length of the MSO formula [Courcelle, 1990].

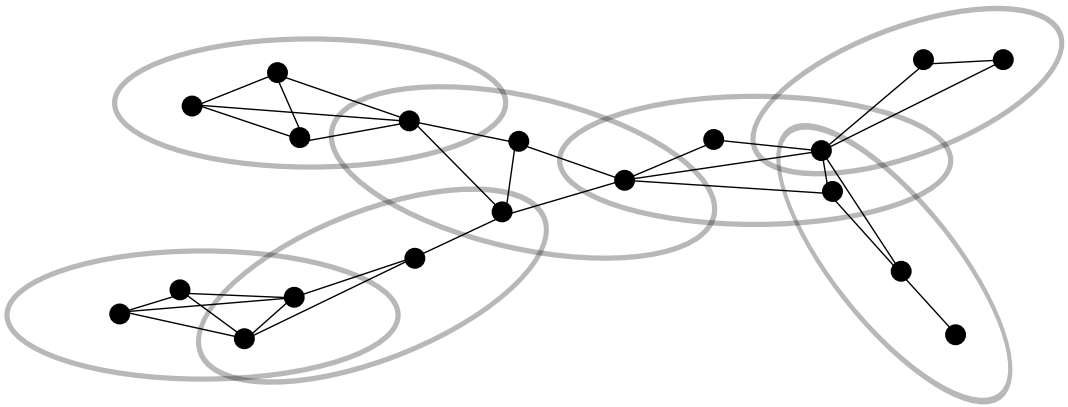
Treewidth: the **King** of Structural Parameters

Fixed parameter tractable (FPT) problems

A **decision problem** with input $\mathcal{I}$ and **parameter** k is FPT parameterized by k if it can be solved in time $f(k) \cdot |\mathcal{I}|^{O(1)}$, where f is a computable function.

Many NP-hard problems are FPT parameterized by treewidth via **dynamic programming** on a **tree-decomposition**: running time $f(\mathtt{tw}) \cdot n^{O(1)}$.

Classic examples: Colouring, Clique, Dominating Set, Hamilton Cycle...



Generally: graph problems expressible in **Monadic Second-Order (MSO) logic** are FPT parameterized by the treewidth plus the length of the MSO formula [Courcelle, 1990].

However, $f(\mathtt{tw})$ may be a tower of exponentials!

ETH-based conditional lower bounds on $f(\mathtt{tw})$ for FPT algorithms

Exponential Time Hypothesis (ETH) [Impagliazzo, Paturi, 1990]

Roughly, n-variable 3-SAT cannot be solved in time $2^{o(n)}$. (stronger than $P \neq NP$)

ETH-based conditional lower bounds on $f(\mathtt{tw})$ for FPT algorithms

Exponential Time Hypothesis (ETH) [Impagliazzo, Paturi, 1990]

Roughly, n-variable 3-SAT cannot be solved in time $2^{o(n)}$. (stronger than $P \neq NP$)

Lower bounds for $f(\mathtt{tw})$ using ETH usually of the form $2^{o(\mathtt{tw})}$, $2^{o(\mathtt{tw}\log\mathtt{tw})}$ or $2^{o(\mathrm{poly}(\mathtt{tw}))}$.

ETH-based conditional lower bounds on $f(\mathtt{tw})$ for FPT algorithms

Exponential Time Hypothesis (ETH) [Impagliazzo, Paturi, 1990]

Roughly, n-variable 3-SAT cannot be solved in time $2^{o(n)}$. (stronger than $P \neq NP$)

Lower bounds for $f(\mathtt{tw})$ using ETH usually of the form $2^{o(\mathtt{tw})}$, $2^{o(\mathtt{tw}\log\mathtt{tw})}$ or $2^{o(\mathrm{poly}(\mathtt{tw}))}$.

Rarer results: Assuming the ETH,

- QSAT (PSPACE-complete) with k alternations admits a lower bound of a tower of exponents of height k in the tw of the primal graph [Fichte, Hecher, Pfandler, 2020];
- k-Choosability (Π_2^p-complete) and k-Choosability Deletion (Σ_3^p-complete) admit double- and triple-exponential lower bounds in tw, resp. [Marx, Mitsou, 2016];
- $\exists\forall$-CSP (Σ_2^p-complete) admits a double-exponential lower bound in the vertex cover number [Lampis, Mitsou, 2017].

ETH-based conditional lower bounds on $f(\mathtt{tw})$ for FPT algorithms

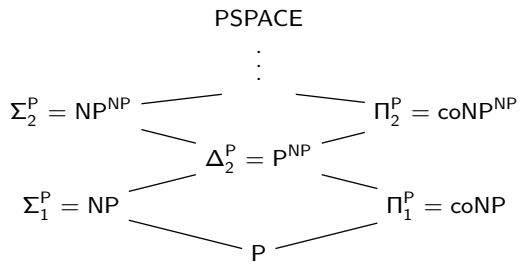
Exponential Time Hypothesis (ETH) [Impagliazzo, Paturi, 1990]

Roughly, n-variable 3-SAT cannot be solved in time $2^{o(n)}$. (stronger than $P \neq NP$)

Lower bounds for $f(\mathtt{tw})$ using ETH usually of the form $2^{o(\mathtt{tw})}$, $2^{o(\mathtt{tw}\log\mathtt{tw})}$ or $2^{o(\mathrm{poly}(\mathtt{tw}))}$.

Rarer results: Assuming the ETH,

- QSAT (PSPACE-complete) with k alternations admits a lower bound of a tower of exponents of height k in the tw of the primal graph [Fichte, Hecher, Pfandler, 2020];
- k-Choosability (Π_2^p-complete) and k-Choosability Deletion (Σ_3^p-complete) admit double- and triple-exponential lower bounds in tw, resp. [Marx, Mitsou, 2016];
- $\exists\forall$-CSP (Σ_2^p-complete) admits a double-exponential lower bound in the vertex cover number [Lampis, Mitsou, 2017].



P: computationally "easy" to solve NP: hard to solve, easy to verify

higher levels: hard to even verify

ETH-based conditional lower bounds on $f(\mathtt{tw})$ for FPT algorithms

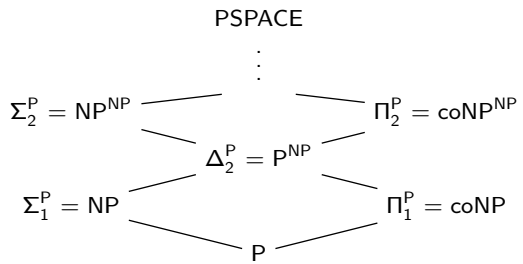
Exponential Time Hypothesis (ETH) [Impagliazzo, Paturi, 1990]

Roughly, n-variable 3-SAT cannot be solved in time $2^{o(n)}$. (stronger than $P \neq NP$)

Lower bounds for $f(\mathtt{tw})$ using ETH usually of the form $2^{o(\mathtt{tw})}$, $2^{o(\mathtt{tw}\log\mathtt{tw})}$ or $2^{o(\mathrm{poly}(\mathtt{tw}))}$.

Rarer results: Assuming the ETH,

- QSAT (PSPACE-complete) with k alternations admits a lower bound of a tower of exponents of height k in the $\mathtt{tw}$ of the primal graph [Fichte, Hecher, Pfandler, 2020];
- k-Choosability (Π_2^p-complete) and k-Choosability Deletion (Σ_3^p-complete) admit double- and triple-exponential lower bounds in $\mathtt{tw}$, resp. [Marx, Mitsou, 2016];
- $\exists\forall$-CSP (Σ_2^p-complete) admits a double-exponential lower bound in the vertex cover number [Lampis, Mitsou, 2017].



P: computationally "easy" to solve NP: hard to solve, easy to verify

higher levels: hard to even verify

Common theme: complexity classes higher than NP require large dependency in $\mathtt{tw}$.

Our results

Main results

ETH-based double-exponential lower bounds in the treewidth for NP-complete problems.

Main results

ETH-based double-exponential lower bounds in the treewidth for NP-complete problems.

We develop a **technique** to prove such lower bounds for (nice) graph problems:

Theorem [F., Galby, Khazaliya, Li, Mc Inerney, Sharma, Tale, ICALP 2024]

Metric Dimension and Geodetic Set

- can be solved in $2^{\mathtt{diam}^{O(\mathtt{tw})}} \cdot n^{O(1)}$ time
- admit no $2^{f(\mathtt{diam})^{o(\mathtt{tw})}} \cdot n^{O(1)}$ time algorithm assuming the ETH

Main results

ETH-based double-exponential lower bounds in the treewidth for NP-complete problems.

We develop a **technique** to prove such lower bounds for (nice) graph problems:

Theorem [F., Galby, Khazaliya, Li, Mc Inerney, Sharma, Tale, ICALP 2024]

Metric Dimension and Geodetic Set

- can be solved in $2^{\mathtt{diam}^{O(\mathtt{tw})}} \cdot n^{O(1)}$ time
- admit no $2^{f(\mathtt{diam})^{o(\mathtt{tw})}} \cdot n^{O(1)}$ time algorithm assuming the ETH

Theorem [Chakraborty, F., Majumdar, Tale, ISAAC 2024]

Locating-Dominating Set and Test Cover

- can be solved in $2^{2^{O(\mathtt{tw})}} \cdot n^{O(1)}$ time
- admit no $2^{2^{o(\mathtt{tw})}} \cdot n^{O(1)}$ time algorithm assuming the ETH

Identification problems

A graph identification problem

Identification problems:

want to **distinguish** all elements of a discrete structure using a (small) solution set

A graph identification problem

Identification problems:

want to **distinguish** all elements of a discrete structure using a (small) solution set

Examples: Test Cover, Separating Set/System, Identifying Code...

$\rightarrow$ studied in combinatorics since the 1960's (Bondy, Rényi...)

A graph identification problem

Identification problems:

want to **distinguish** all elements of a discrete structure using a (small) solution set

Examples: Test Cover, Separating Set/System, Identifying Code...

$\rightarrow$ studied in combinatorics since the 1960's (Bondy, Rényi...)

Locating-dominating set [Slater, 1980's]

$D \subseteq V$ **locating-dominating set** of $G = (V, E)$:

- for every $v \in V$, $N[v] \cap D \neq \emptyset$ (domination)
- $\forall u \neq v$ of $V \setminus D$, $N(u) \cap D \neq N(v) \cap D$ (location)

(Every vertex not in D has a distinct neighbourhood within the solution D)

A graph identification problem

Identification problems:

want to **distinguish** all elements of a discrete structure using a (small) solution set

Examples: Test Cover, Separating Set/System, Identifying Code...

$\rightarrow$ studied in combinatorics since the 1960's (Bondy, Rényi...)

Locating-dominating set [Slater, 1980's]

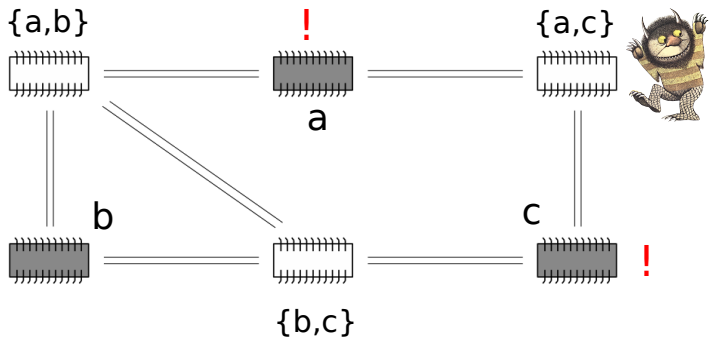
$D \subseteq V$ **locating-dominating set** of $G = (V, E)$:

- for every $v \in V$, $N[v] \cap D \neq \emptyset$ (domination)
- $\forall u \neq v$ of $V \setminus D$, $N(u) \cap D \neq N(v) \cap D$ (location)

(Every vertex not in D has a distinct neighbourhood within the solution D)

Motivation: fault-detection in networks.

$\rightarrow$ The set D of grey processors is a set of fault-detectors.



A graph identification problem

Identification problems:

want to **distinguish** all elements of a discrete structure using a (small) solution set

Examples: Test Cover, Separating Set/System, Identifying Code...

→ studied in combinatorics since the 1960's (Bondy, Rényi...)

Locating-dominating set [Slater, 1980's]

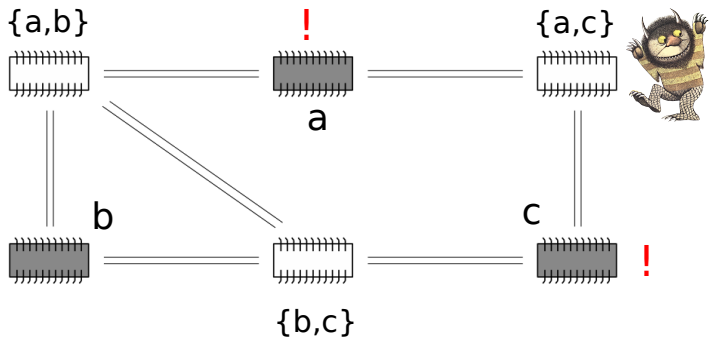
$D \subseteq V$ **locating-dominating set** of $G = (V, E)$:

- for every $v \in V$, $N[v] \cap D \neq \emptyset$ (domination)
- $\forall u \neq v$ of $V \setminus D$, $N(u) \cap D \neq N(v) \cap D$ (location)

(Every vertex not in D has a distinct neighbourhood within the solution D)

Motivation: fault-detection in networks.

→ The set D of grey processors is a set of fault-detectors.



Other applications:

biological testing, graph isomorphism, machine learning (VC-dimension)...

A graph identification problem

Identification problems:

want to **distinguish** all elements of a discrete structure using a (small) solution set

Examples: Test Cover, Separating Set/System, Identifying Code...

$\rightarrow$ studied in combinatorics since the 1960's (Bondy, Rényi...)

Locating-dominating set [Slater, 1980's]

$D \subseteq V$ **locating-dominating set** of $G = (V, E)$:

- for every $v \in V$, $N[v] \cap D \neq \emptyset$ (domination)
- $\forall u \neq v$ of $V \setminus D$, $N(u) \cap D \neq N(v) \cap D$ (location)

(Every vertex not in D has a distinct neighbourhood within the solution D)

Decision problem:

Locating-Dominating Set

Input: an undirected graph $G = (V, E)$ and an integer $k \geq 1$
Question: Does G have a locating-dominating set of size at most k?

NP-complete, FPT for treewidth using MSOL and Courcelle's theorem

Proof ideas

Main result for Locating-dominating Set

Theorem [Chakraborty, F., Majumdar, Tale, ISAAC 2024]

Locating-Dominating Set

- can be solved in $2^{2^{O(\mathtt{tw})}} \cdot n^{O(1)}$ time
- admits no $2^{2^{o(\mathtt{tw})}} \cdot n^{O(1)}$ time algorithm assuming the ETH

Main result for Locating-dominating Set

Theorem [Chakraborty, F., Majumdar, Tale, ISAAC 2024]

Locating-Dominating Set

- can be solved in $2^{2^{O(\mathtt{tw})}} \cdot n^{O(1)}$ time
- admits no $2^{2^{o(\mathtt{tw})}} \cdot n^{O(1)}$ time algorithm assuming the ETH

Algorithm: classic **dynamic programming** over a tree-decomposition.

Main idea: for each bag, remember which **subsets** of solution vertices form the intersection of some neighborhood

$\rightarrow$ Number of DP-states per bag: $2^{2^{O(\mathtt{tw})}}$ possible collections of subsets

Main result for Locating-dominating Set: lower bound

Locating-Dominating Set admits no $2^{2^{o(\mathtt{tw})}} \cdot n^{O(1)}$ time algorithm assuming the ETH.

Exponential Time Hypothesis (ETH) [Impagliazzo, Paturi, 1990]

Roughly, n-variable 3-SAT cannot be solved in time $2^{o(n)}$.

Goal: reduction from n-variable 3-SAT such that $\mathtt{tw}(G) = O(\log n)$.

$\rightarrow$ Algorithm running in $2^{2^{o(\mathtt{tw})}} = 2^{2^{o(\log n)}} = 2^{o(n)}$ $\rightarrow$ contradicts the ETH.

Main result for Locating-dominating Set: lower bound

Locating-Dominating Set admits no $2^{2^{o(\mathtt{tw})}} \cdot n^{O(1)}$ time algorithm assuming the ETH.

Exponential Time Hypothesis (ETH) [Impagliazzo, Paturi, 1990]

Roughly, n-variable 3-SAT cannot be solved in time $2^{o(n)}$.

Goal: reduction from n-variable 3-SAT such that $\mathtt{tw}(G) = O(\log n)$.

$\rightarrow$ Algorithm running in $2^{2^{o(\mathtt{tw})}} = 2^{2^{o(\log n)}} = 2^{o(n)}$ $\rightarrow$ contradicts the ETH.

Fact: can assume every variable appears ≤ 3 times (standard reduction)

Main result for Locating-dominating Set: lower bound

Locating-Dominating Set admits no $2^{2^{o(\mathtt{tw})}} \cdot n^{O(1)}$ time algorithm assuming the ETH.

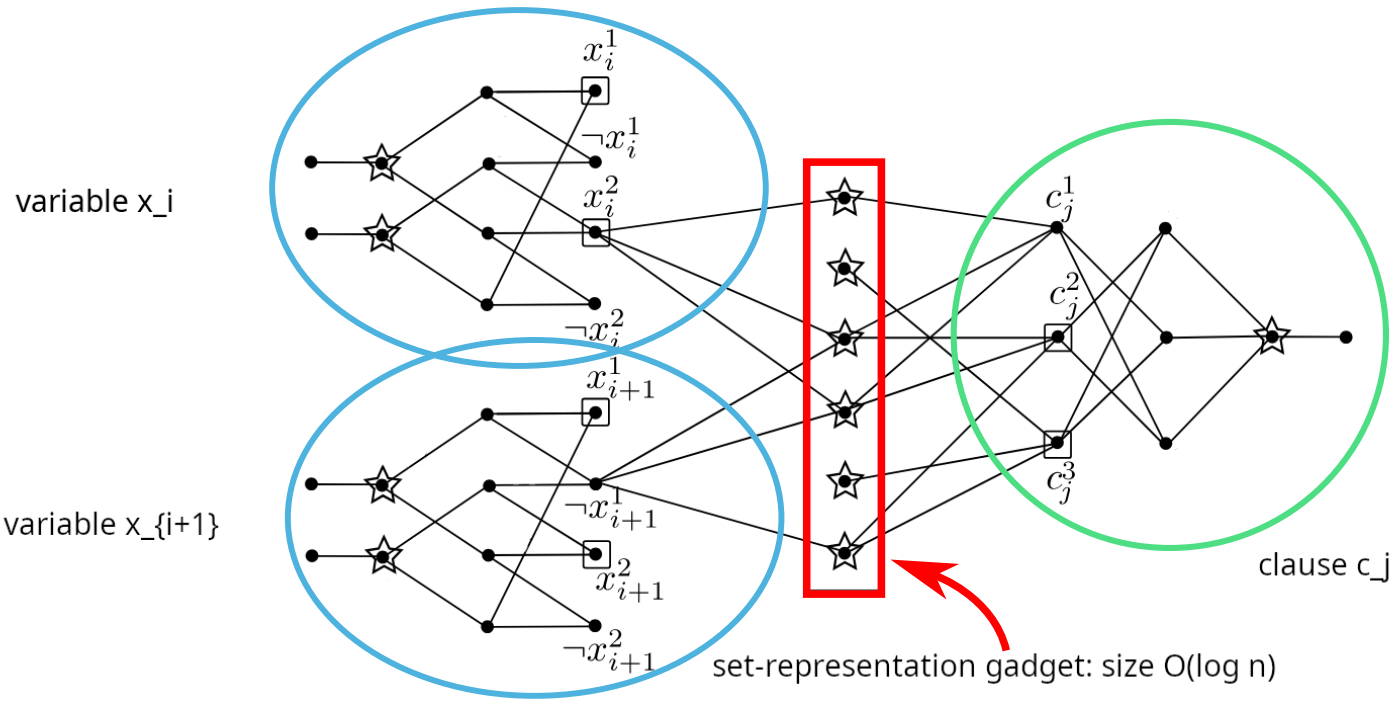
Exponential Time Hypothesis (ETH) [Impagliazzo, Paturi, 1990]

Roughly, n-variable 3-SAT cannot be solved in time $2^{o(n)}$.

Goal: reduction from n-variable 3-SAT such that $\mathtt{tw}(G) = O(\log n)$.

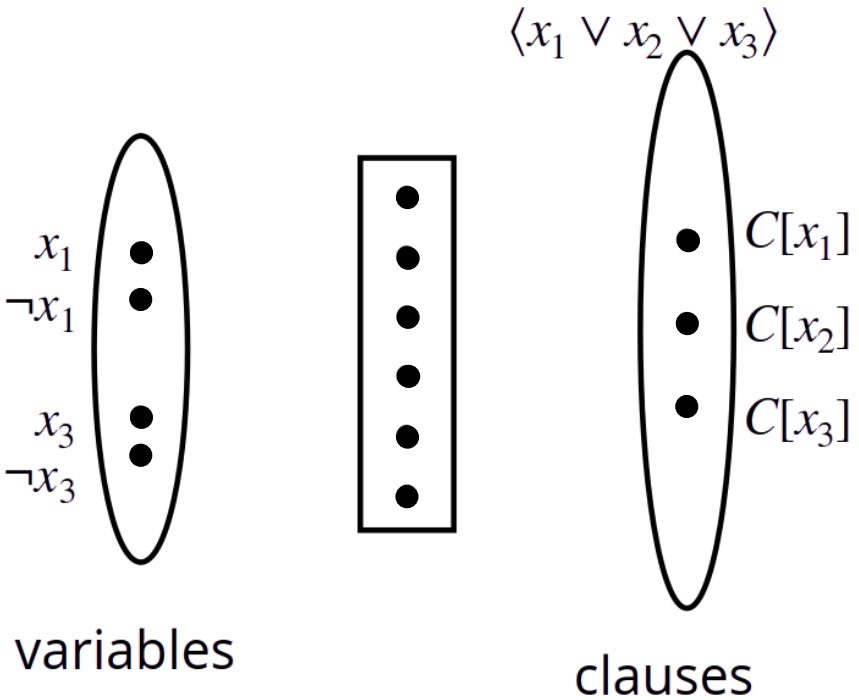
$\rightarrow$ Algorithm running in $2^{2^{o(\mathtt{tw})}} = 2^{2^{o(\log n)}} = 2^{o(n)} \rightarrow$ contradicts the ETH.

Fact: can assume every variable appears ≤ 3 times (standard reduction)



How to encode clause-variable relations using a set of size $O(\log n)$?

Set-representation gadget: vertex set S of size $2p$.

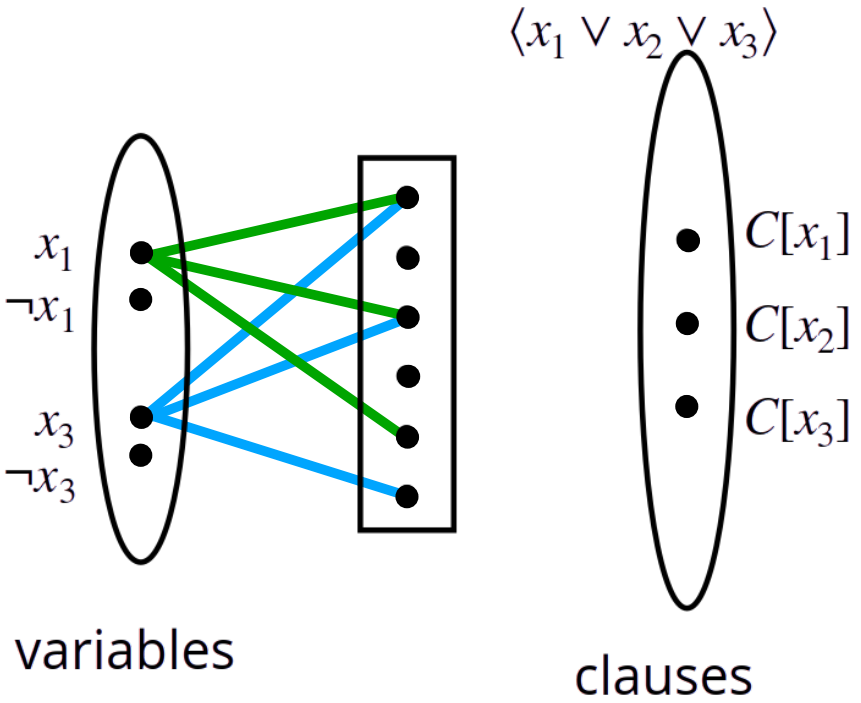


$S_1 = \{1,3,5\}$ $S_3 = \{1,3,6\}$

How to encode clause-variable relations using a set of size $O(\log n)$?

Set-representation gadget: vertex set S of size $2p$. Assign as neighbours, a **distinct subset** of size p to **each literal occurrence**. $\binom{2p}{p} \geq 4n \rightarrow p = O(\log n)$

(these subsets form a Sperner family)



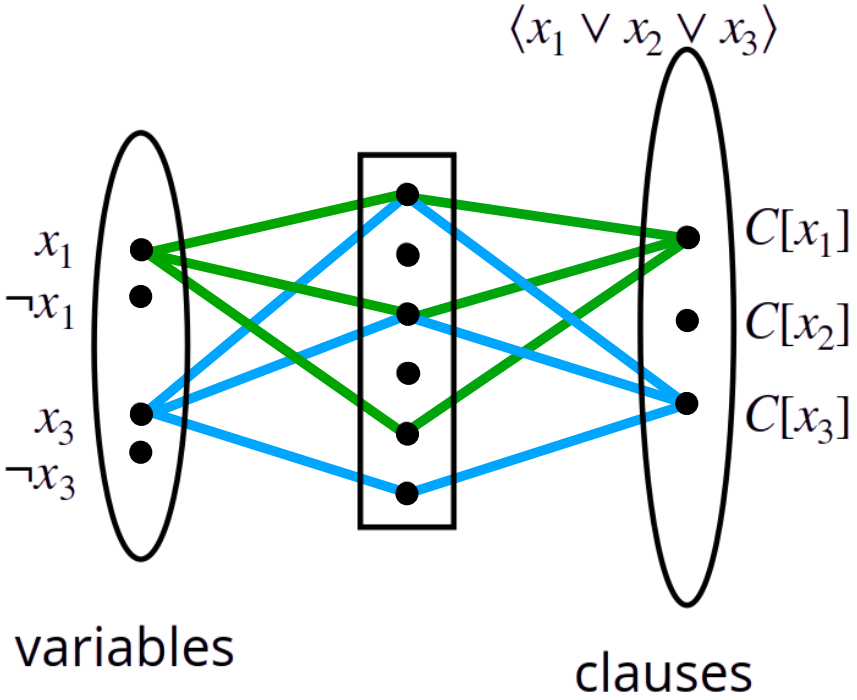
$S_1 = \{1,3,5\}$ $S_3 = \{1,3,6\}$

How to encode clause-variable relations using a set of size $O(\log n)$?

Set-representation gadget: vertex set S of size $2p$. Assign as neighbours, a **distinct subset** of size p to **each literal occurrence**. $\binom{2p}{p} \geq 4n \rightarrow p = O(\log n)$

(these subsets form a Sperner family)

Each clause C is represented by 3 vertices, each connected to the same vertices in S as one of the literals in C.



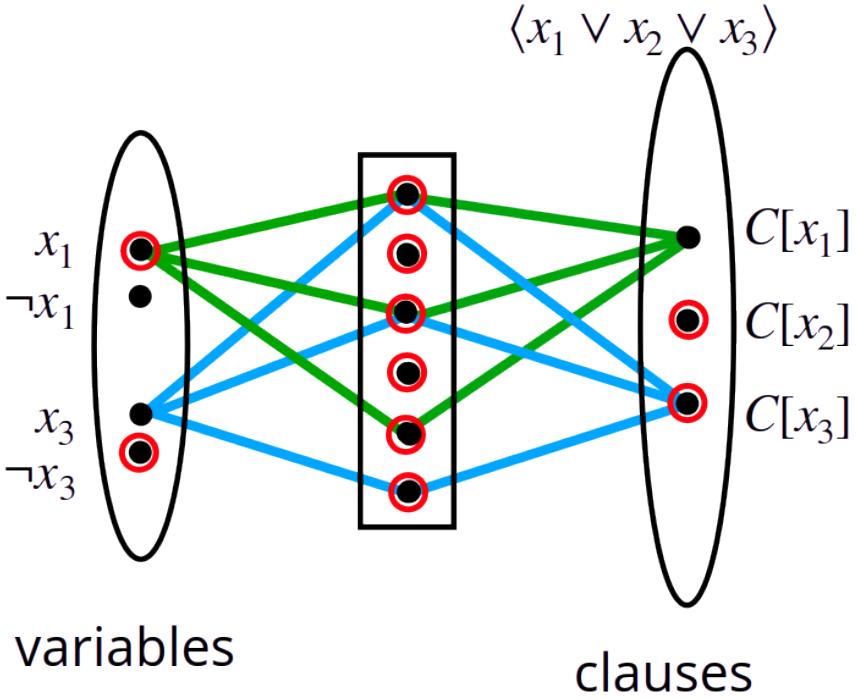
$S_1 = \{1,3,5\}$ $S_3 = \{1,3,6\}$

How to encode clause-variable relations using a set of size $O(\log n)$?

Set-representation gadget: vertex set S of size $2p$. Assign as neighbours, a **distinct subset** of size p to **each literal occurrence**. $\binom{2p}{p} \geq 4n \rightarrow p = O(\log n)$

(these subsets form a Sperner family)

Each clause C is represented by 3 vertices, each connected to the same vertices in S as one of the literals in C.



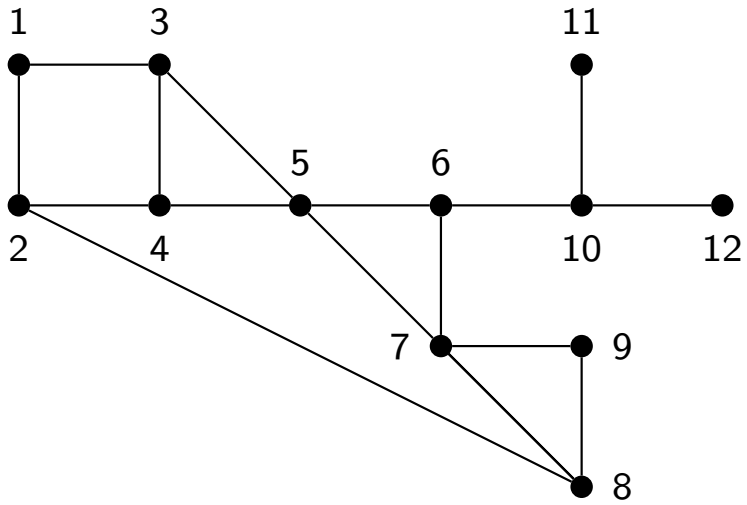
$S_1 = \{1,3,5\}$ $S_3 = \{1,3,6\}$

Metric Dimension

Metric dimension

Metric dimension of a graph $G = (V, E)$ [Slater '75 + Harary, Melter '76]

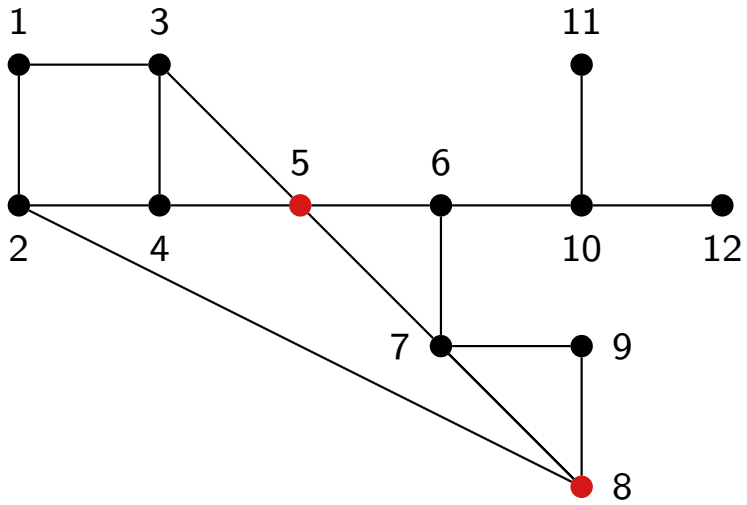
$S \subseteq V$ is a resolving set of G if $\forall u, v \in V$, $\exists z \in S$ with $d(z, u) \neq d(z, v)$. The **minimum size** of a resolving set of G is the metric dimension of G.



Metric dimension

Metric dimension of a graph $G = (V, E)$ [Slater '75 + Harary, Melter '76]

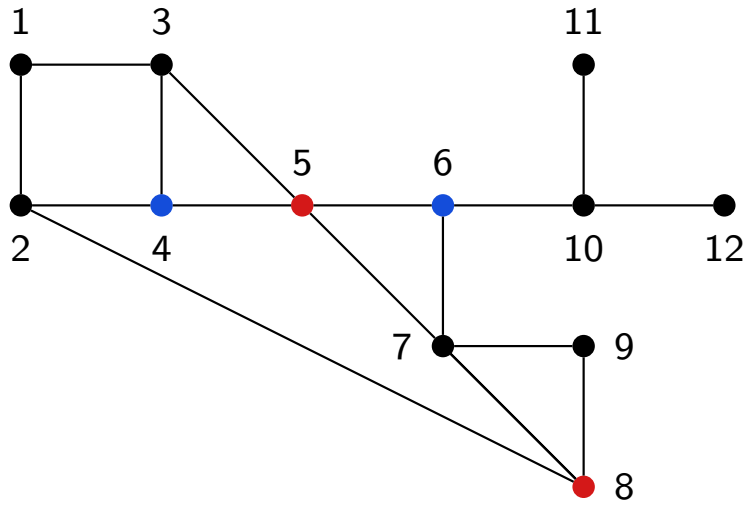
$S \subseteq V$ is a resolving set of G if $\forall u, v \in V$, $\exists z \in S$ with $d(z, u) \neq d(z, v)$. The **minimum size** of a resolving set of G is the metric dimension of G.



Metric dimension

Metric dimension of a graph $G = (V, E)$ [Slater '75 + Harary, Melter '76]

$S \subseteq V$ is a resolving set of G if $\forall u, v \in V$, $\exists z \in S$ with $d(z, u) \neq d(z, v)$. The **minimum size** of a resolving set of G is the metric dimension of G.

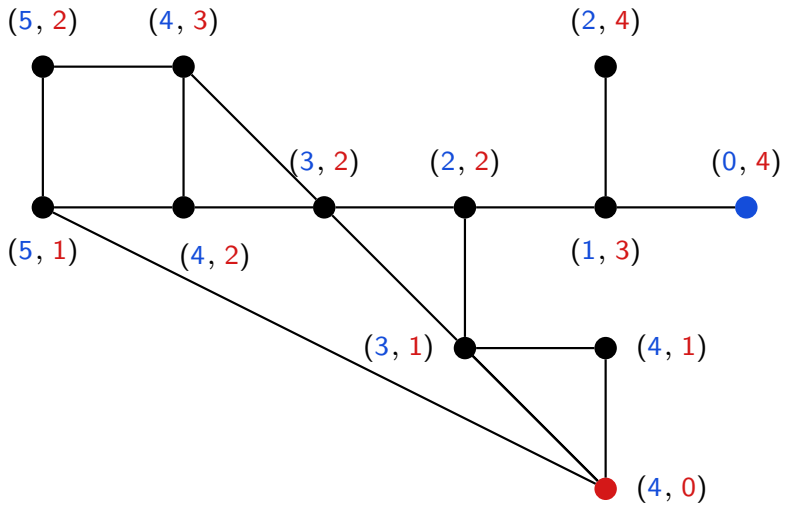


Vertices 4 and 6 are **not** resolved by 5 nor 8.

Metric dimension

Metric dimension of a graph $G = (V, E)$ [Slater '75 + Harary, Melter '76]

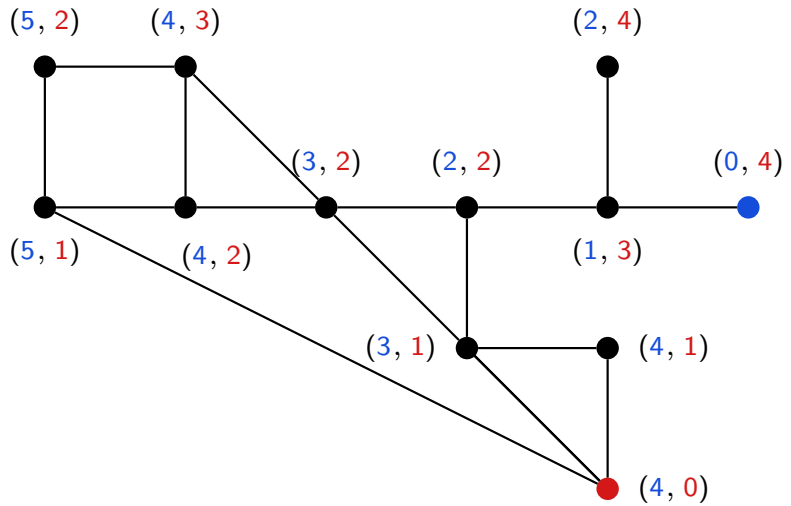
$S \subseteq V$ is a resolving set of G if $\forall u, v \in V$, $\exists z \in S$ with $d(z, u) \neq d(z, v)$. The **minimum size** of a resolving set of G is the metric dimension of G.



Metric dimension

Metric dimension of a graph $G = (V, E)$ [Slater '75 + Harary, Melter '76]

$S \subseteq V$ is a resolving set of G if $\forall u, v \in V$, $\exists z \in S$ with $d(z, u) \neq d(z, v)$. The **minimum size** of a resolving set of G is the metric dimension of G.



Metric Dimension

Input: an undirected graph $G = (V, E)$ and an integer $k \geq 1$
Question: Is the metric dimension of G at most k?

Results for Metric Dimension

Metric Dimension is NP-hard for graphs of treewidth 24 (Li-Pilipczuk, 2021)

But: becomes FPT for treewidth if the diameter is bounded.

Results for Metric Dimension

Metric Dimension is NP-hard for graphs of treewidth 24 (Li-Pilipczuk, 2021)

But: becomes FPT for treewidth if the diameter is bounded.

Theorem [F., Galby, Khazaliya, Li, Mc Inerney, Sharma, Tale, ICALP 2024]

Metric Dimension:

- can be solved in $2^{\mathtt{diam}^{O(\mathtt{tw})}} \cdot n^{O(1)}$ time
- admits no $2^{f(\mathtt{diam})^{o(\mathtt{tw})}} \cdot n^{O(1)}$ time algorithm assuming the ETH

Results for Metric Dimension

Metric Dimension is NP-hard for graphs of treewidth 24 (Li-Pilipczuk, 2021)

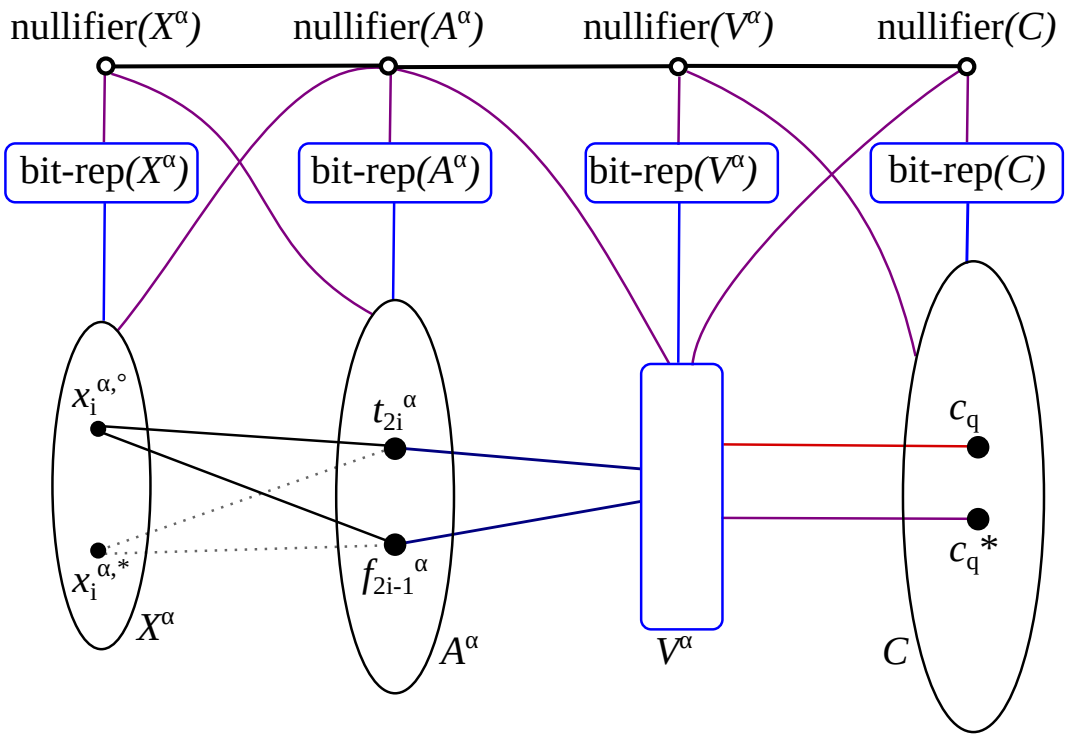
But: becomes FPT for treewidth if the diameter is bounded.

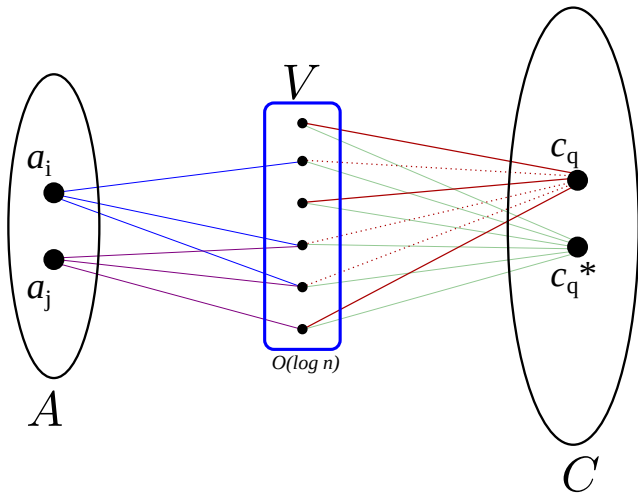
Theorem [F., Galby, Khazaliya, Li, Mc Inerney, Sharma, Tale, ICALP 2024]

Metric Dimension:

- can be solved in $2^{\mathtt{diam}^{O(\mathtt{tw})}} \cdot n^{O(1)}$ time
- admits no $2^{f(\mathtt{diam})^{o(\mathtt{tw})}} \cdot n^{O(1)}$ time algorithm assuming the ETH

The original proof is a more complex use-case of the set-representation gadget:





Conclusion

Main takeaway:

- Most natural NP-complete problems FPT for $\mathtt{tw}$ can be solved in time $2^{poly(\mathtt{tw})} \cdot n^{O(1)}$
- **Graph identification problems** require $2^{2^{O(\mathtt{tw})}} \cdot n^{O(1)}$ time, assuming the ETH

Conclusion

Main takeaway:

- Most natural NP-complete problems FPT for $\mathtt{tw}$ can be solved in time $2^{poly(\mathtt{tw})} \cdot n^{O(1)}$
- **Graph identification problems** require $2^{2^{O(\mathtt{tw})}} \cdot n^{O(1)}$ time, assuming the ETH

Further results:

- These problems have other "exotic" behaviours
- Other NP-complete problems behave similarly:
  Geodetic Set (on graphs of bounded diameter)
- Others admit lower bounds for the larger **vertex cover number**:
  Strong Metric Dimension

Conclusion

Main takeaway:

- Most natural NP-complete problems FPT for $\mathtt{tw}$ can be solved in time $2^{poly(\mathtt{tw})} \cdot n^{O(1)}$
- **Graph identification problems** require $2^{2^{O(\mathtt{tw})}} \cdot n^{O(1)}$ time, assuming the ETH

Further results:

- These problems have other "exotic" behaviours
- Other NP-complete problems behave similarly:
  Geodetic Set (on graphs of bounded diameter)
- Others admit lower bounds for the larger **vertex cover number**:
  Strong Metric Dimension

Identification problems:

- Have intriguing algorithmic behaviours
- Also interesting from graph-theoretical point of view
- Nice, sometimes challenging, research problems and solutions

Conclusion

Main takeaway:

- Most natural NP-complete problems FPT for $\mathtt{tw}$ can be solved in time $2^{poly(\mathtt{tw})} \cdot n^{O(1)}$
- **Graph identification problems** require $2^{2^{O(\mathtt{tw})}} \cdot n^{O(1)}$ time, assuming the ETH

Further results:

- These problems have other "exotic" behaviours
- Other NP-complete problems behave similarly:
  Geodetic Set (on graphs of bounded diameter)
- Others admit lower bounds for the larger **vertex cover number**:
  Strong Metric Dimension

Identification problems:

- Have intriguing algorithmic behaviours
- Also interesting from graph-theoretical point of view
- Nice, sometimes challenging, research problems and solutions

Question: what about **triple-exponential** lower bounds for natural NP-complete problems?

Conclusion

Main takeaway:

- Most natural NP-complete problems FPT for $\mathtt{tw}$ can be solved in time $2^{poly(\mathtt{tw})} \cdot n^{O(1)}$
- **Graph identification problems** require $2^{2^{O(\mathtt{tw})}} \cdot n^{O(1)}$ time, assuming the ETH

Further results:

- These problems have other "exotic" behaviours
- Other NP-complete problems behave similarly:
  Geodetic Set (on graphs of bounded diameter)
- Others admit lower bounds for the larger **vertex cover number**:
  Strong Metric Dimension

Identification problems:

- Have intriguing algorithmic behaviours
- Also interesting from graph-theoretical point of view
- Nice, sometimes challenging, research problems and solutions

Question: what about **triple-exponential** lower bounds for natural NP-complete problems?

THANKS!