CSPs for children: Fine-grained complexity of graph homomorphism problems

Paweł Rzążewski

Warsaw University of Technology & University of Warsaw

Graph homomorphism: adult's view

- Graph homomorphism $\equiv$ finite-domain CSP with one binary symmetric relation

Graph homomorphism: adult's view

- Graph homomorphism $\equiv$ finite-domain CSP with one binary symmetric relation

- only one? and symmetric?

Why should we care about CSPs for children?

Even if meant for kids, still fun for adults

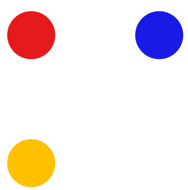
Children like coloring...

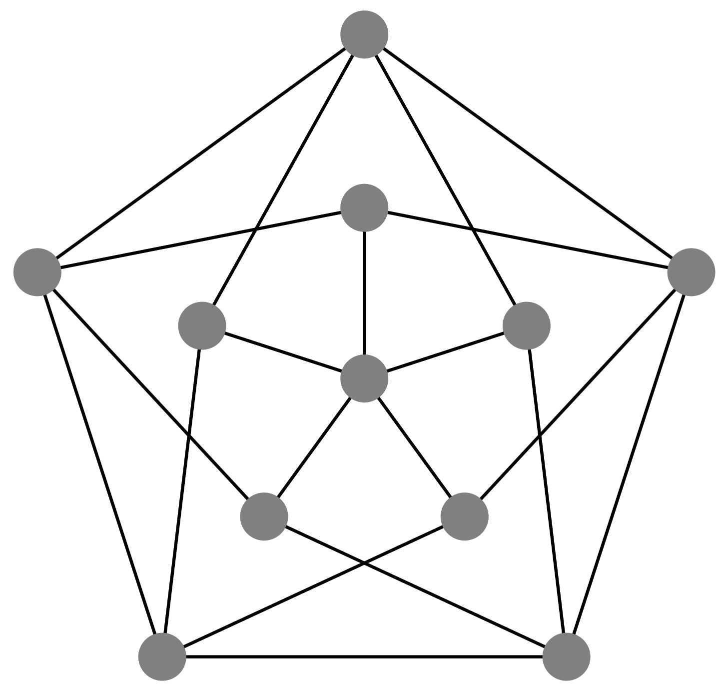
k-Coloring

Input: a graph G with n vertices

Question: can G be properly colored with k colors?

$k = 3$





No

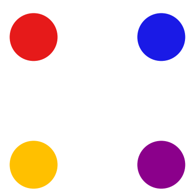
Children like coloring...

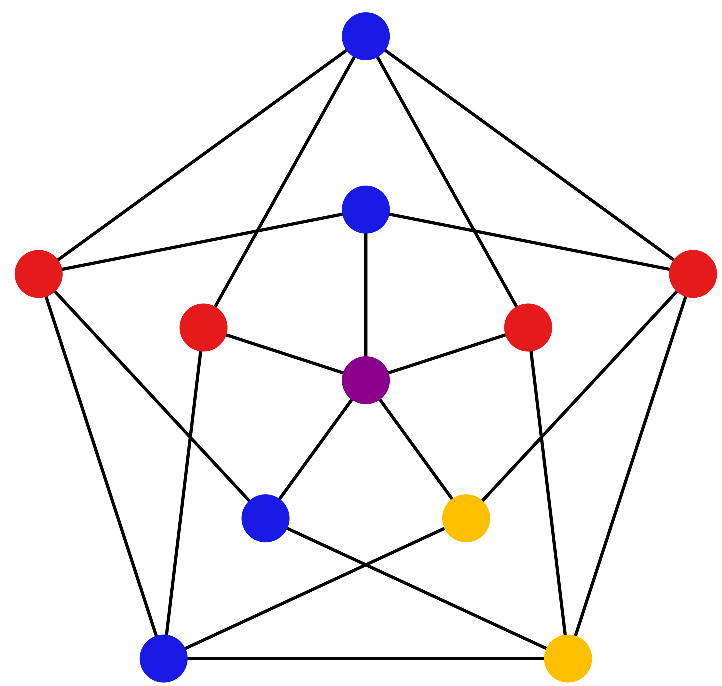
k-Coloring

Input: a graph G with n vertices

Question: can G be properly colored with k colors?

$k = 4$





Yes

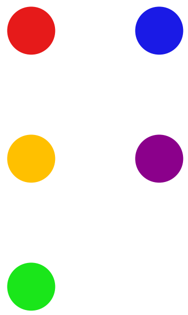
Children like coloring...

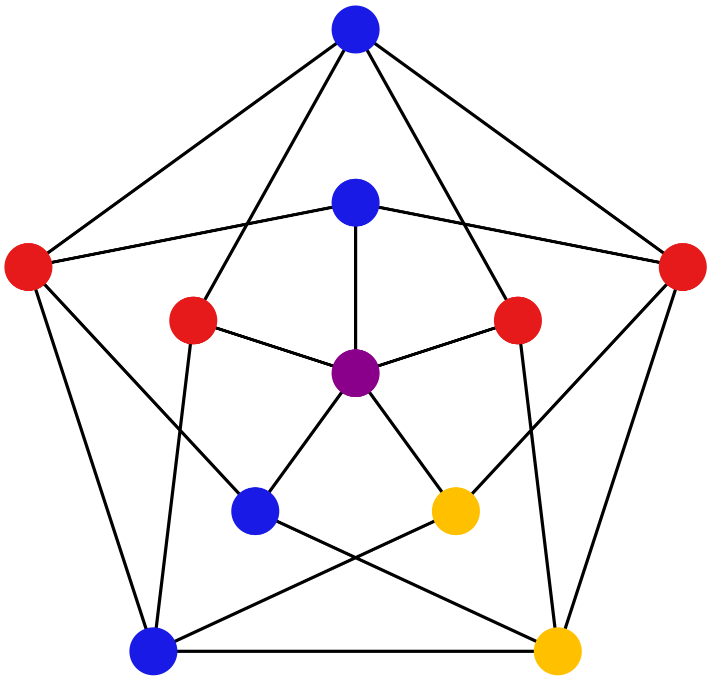
k-Coloring

Input: a graph G with n vertices

Question: can G be properly colored with k colors?

$k = 5$





Yes

Some classics

Theorem (Szekeres-Wilf). If every subgraph of G has a vertex of degree $\leqslant k-1$, then G admits a proper k-coloring.

Theorem (Appel, Haken). Every planar graph admits a proper 4-coloring.

Some classics

Theorem (Szekeres-Wilf). If every subgraph of G has a vertex of degree $\leqslant k-1$, then G admits a proper k-coloring.

Theorem (Appel, Haken). Every planar graph admits a proper 4-coloring.

Theorem (Karp). For every $k \geqslant 3$, the k-Coloring problem is NP-complete (and polynomial-time-solvable for $k \leqslant 2$).

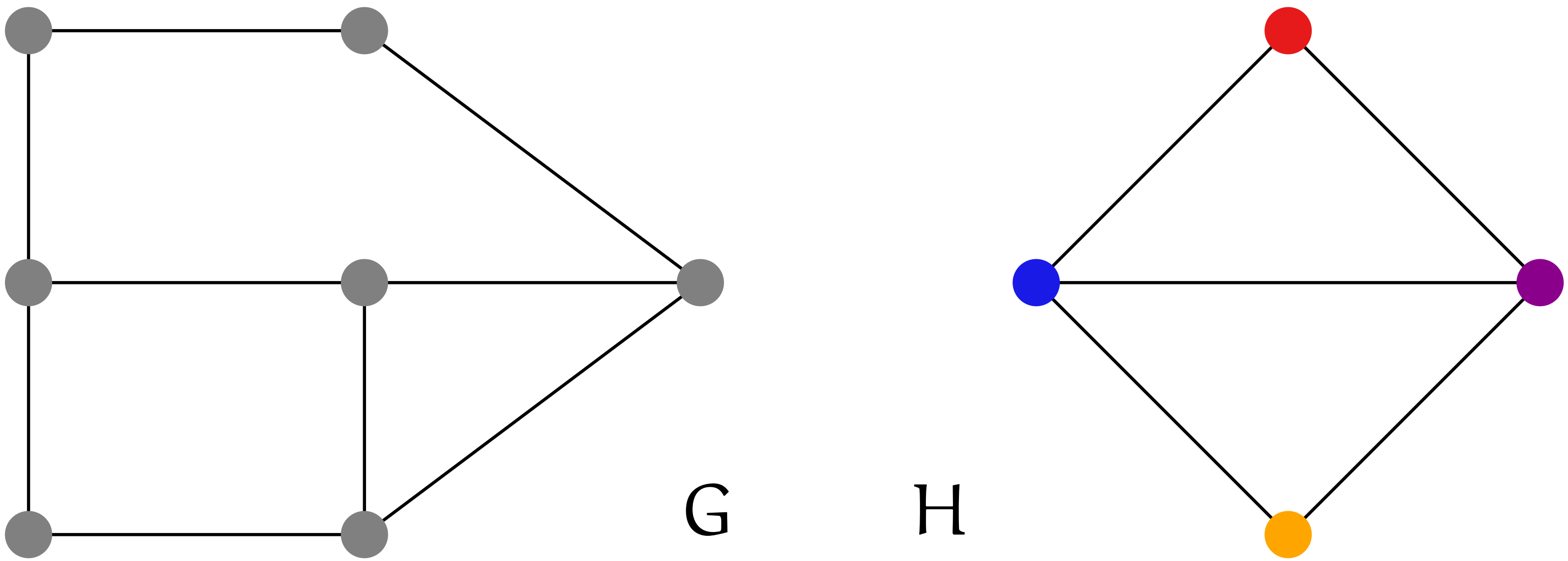
- brute force: $k^n \cdot \mathrm{poly}(n)$

Theorem (Björklund, Husfeldt, Koivisto). For every k, the k-Coloring problem can be solved in time $2^n \cdot \mathrm{poly}(n)$.

not depending on k

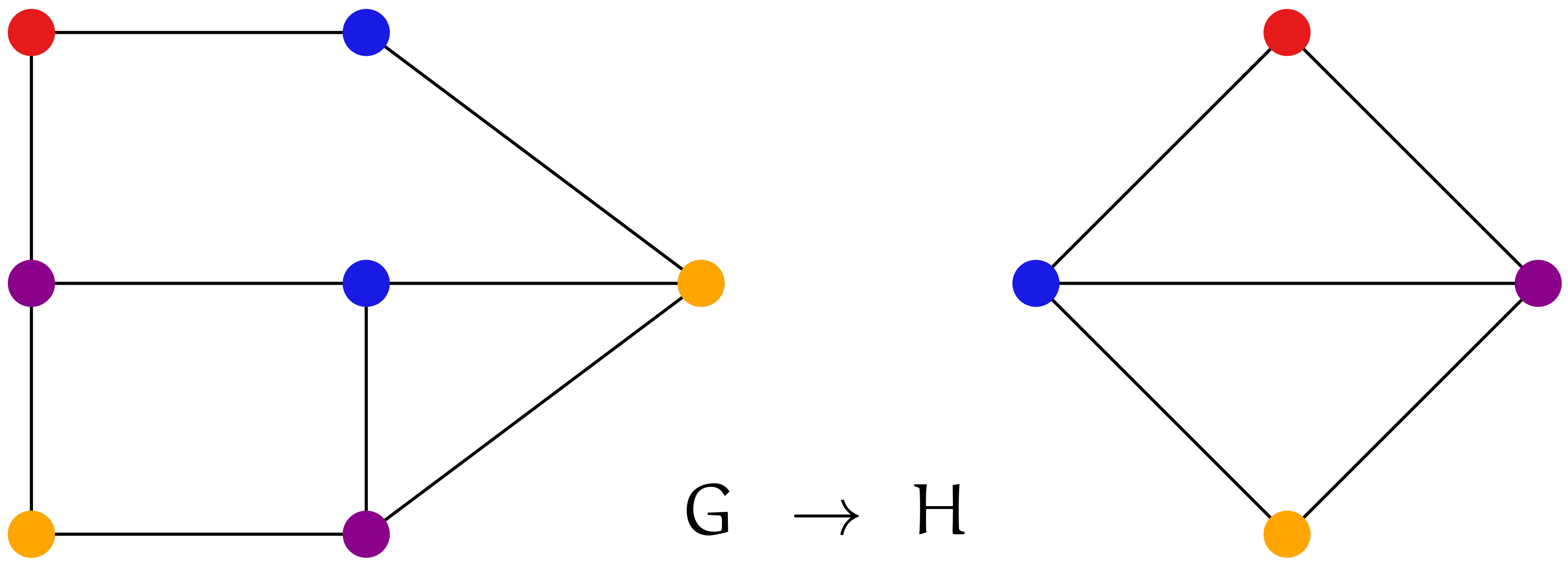
Graph coloring for grown-ups

Homomorphism from G to H (also called an H-coloring) $\equiv$ edge-preserving mapping from $V(G)$ to $V(H)$



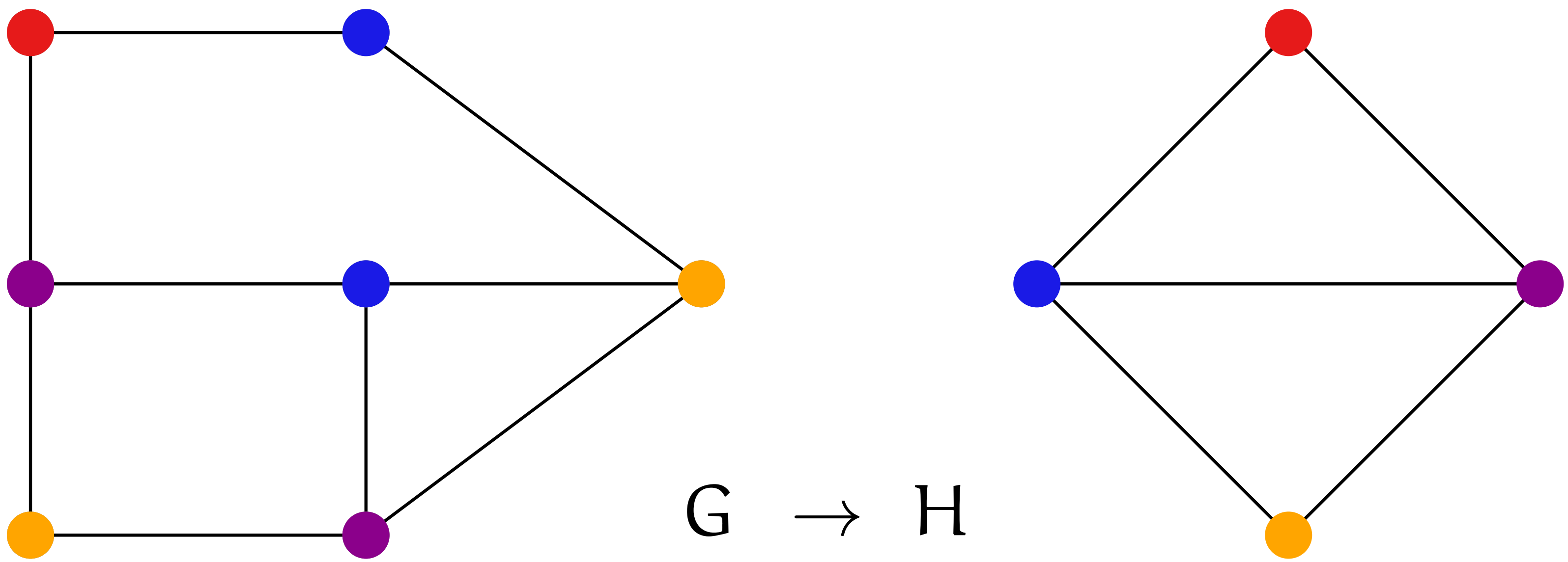
Graph coloring for grown-ups

Homomorphism from G to H (also called an H-coloring) $\equiv$ edge-preserving mapping from $V(G)$ to $V(H)$



Graph coloring for grown-ups

Homomorphism from G to H (also called an H-coloring) $\equiv$ edge-preserving mapping from $V(G)$ to $V(H)$



H-Coloring

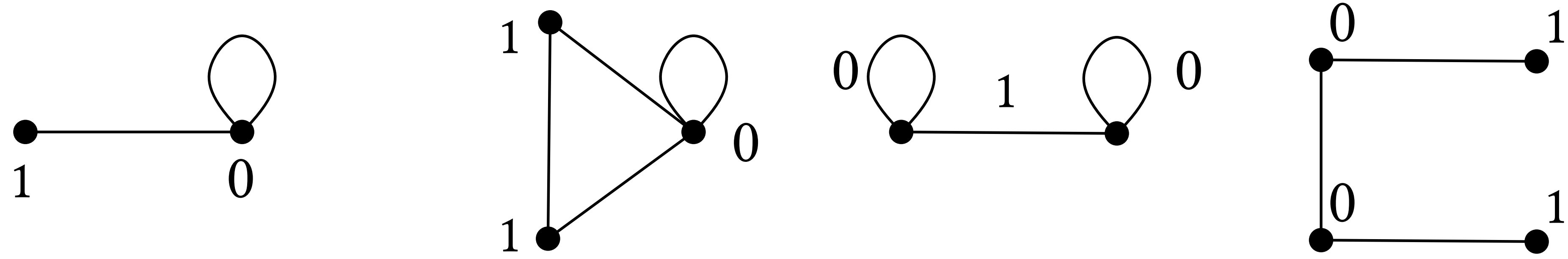
Input: a graph G with n vertices

Question: does G admit an H-coloring?

- K_k-Coloring $\equiv$ k-Coloring

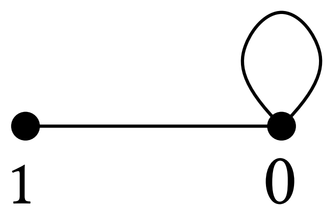
Not only colorings

maximize total weight

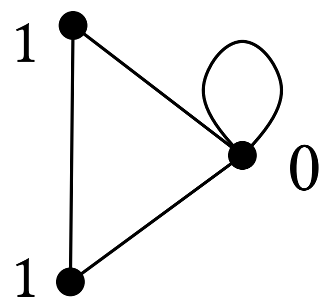


Not only colorings

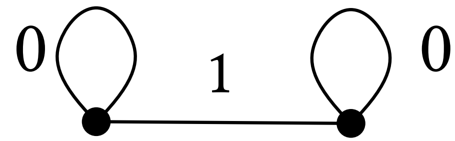
maximize total weight



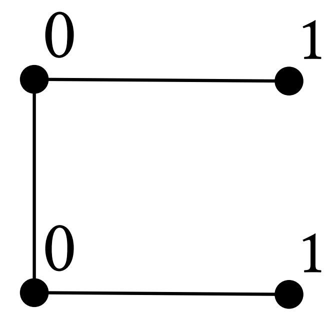
Independent Set



Odd Cycle Transversal



Max Cut

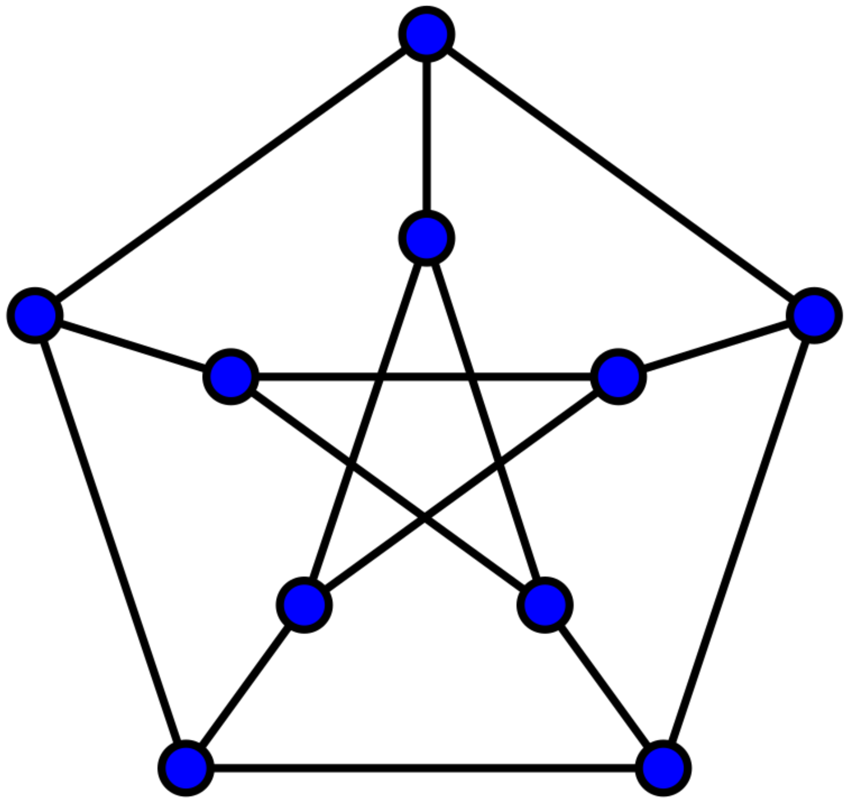


Independent Set in bipartite graphs

Classics rediscovered

Theorem (Szekeres-Wilf). If every subgraph of G has a vertex of degree $\leqslant k-1$, then G admits a proper k-coloring.

Theorem (Chen, Raspaud). If every subgraph of G has average degree < 2.5 and G has no triangles, then G admits a homomorphism to the Petersen graph.



- first case of a very nice conjecture!

Classics rediscovered, ctd.

Theorem (Appel, Haken). Every planar graph admits a proper 4-coloring.

Theorem (Grötzsch). Every triangle-free planar graph admits a proper 3-coloring.

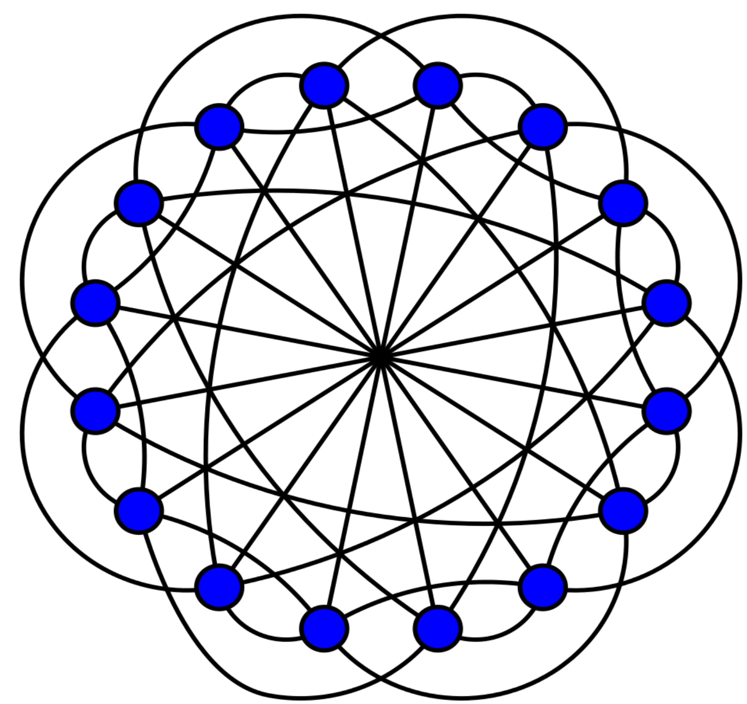
Classics rediscovered, ctd.

Theorem (Appel, Haken). Every planar graph admits a proper 4-coloring.

Theorem (Grötzsch). Every triangle-free planar graph admits a proper 3-coloring.

Theorem (Naserasr, Migussie, Škrekovski). Every triangle-free planar graph admits a homomorphism to the Clebsch graph.

it’s planar and triangle-free



Complexity of the problem

Theorem (Karp).
For every $k \geqslant 3$, the k-Coloring problem is NP-complete.

Theorem (Hell, Nešetřil).
For every loopless, nonbipartite H, the H-Coloring problem is NP-complete.

- otherwise is polynomial-time-solvable (and easy)

Complexity of the problem

Theorem (Karp).
For every $k \geqslant 3$, the k-Coloring problem is NP-complete.

Theorem (Hell, Nešetřil).
For every loopless, nonbipartite H, the H-Coloring problem is NP-complete.

- otherwise is polynomial-time-solvable (and easy)

Theorem (Björklund, Husfeldt, Koivisto).
For every k, the k-Coloring problem can be solved in time $2^n \cdot \mathrm{poly}(n)$.

Theorem (Cygan, Fomin, Golovnev, Kulikov, Mihajlin, Pachocki, Socała).
There is no $2^{o(n \log |H|)}$ algorithm, assuming the ETH.

- no c^n-algorithm for universal constant c

Coloring bounded-treewidth graphs

- From now on assume that G has n vertices and is given with a tree decomposition of width tw
- For every k, k-Coloring can be decided in time $k^{\mathsf{tw}} \cdot \mathsf{poly}(n)$
- same for list coloring, for counting colorings...

Coloring bounded-treewidth graphs

- From now on assume that G has n vertices and is given with a tree decomposition of width tw
- For every k, k-Coloring can be decided in time $k^{\mathrm{tw}} \cdot \mathrm{poly}(n)$
- same for list coloring, for counting colorings...

Theorem (Lokshtanov, Marx, Saubrabh).
For any $k \geqslant 3$, k-Coloring cannot be solved in time $(k - \varepsilon)^{\mathrm{tw}} \cdot \mathrm{poly}(n)$, assuming the SETH.

- ... and thus also list coloring, counting colorings etc.

Homomorphisms and bounded treewidth

- we consider connected non-bipartite graphs H with no loops
- we can also assume that H is a *core* (has no homomorphism to its proper subgraph)

Problem.

For every graph H, find $k = k(H)$, such that H-coloring of G

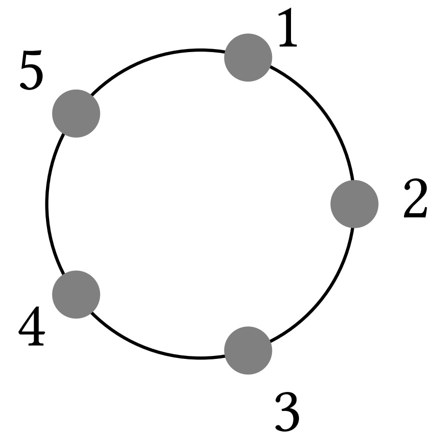
- can be solved in time k^{tw},
- cannot be solved in time $(k - \varepsilon)^{tw}$, unless the SETH fails.

let's not write $\cdot\text{poly}(n)$

- $k \leqslant |H|$
- if H is complete, then $k = |H|$

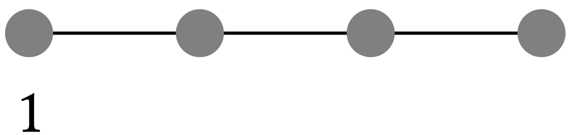
Warm-up: C_5

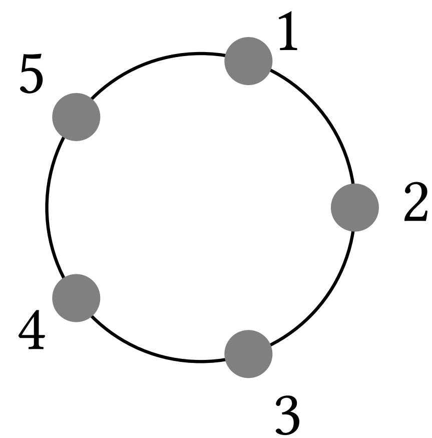
- the smallest non-complete core
- can you beat 5^{tw}?



Warm-up: C_5

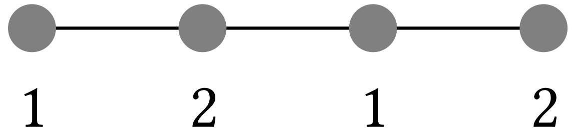
- the smallest non-complete core
- can you beat 5^{tw}?

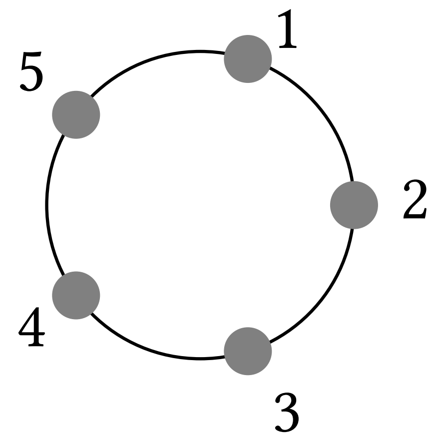




Warm-up: C_5

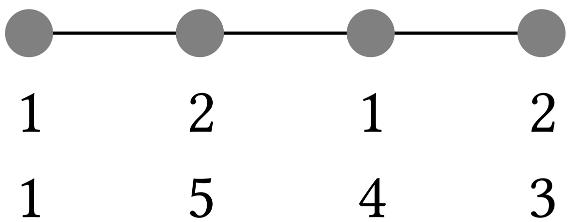
- the smallest non-complete core
- can you beat 5^{tw}?

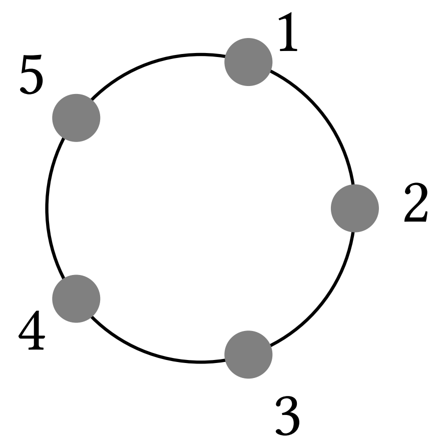




Warm-up: C_5

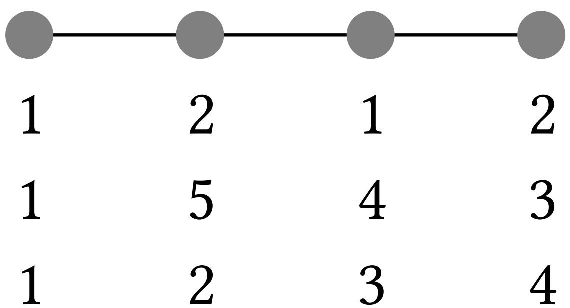
- the smallest non-complete core
- can you beat 5^{tw}?

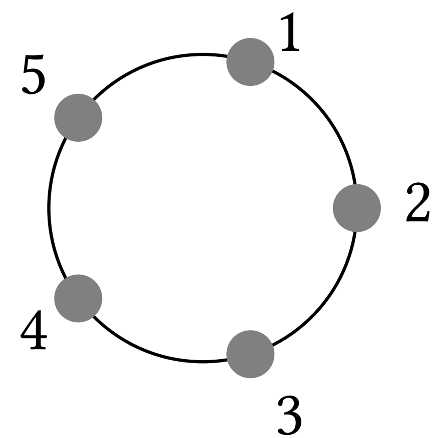




Warm-up: C_5

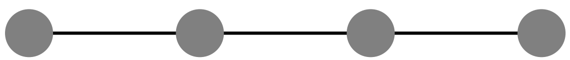
- the smallest non-complete core
- can you beat 5^{tw}?



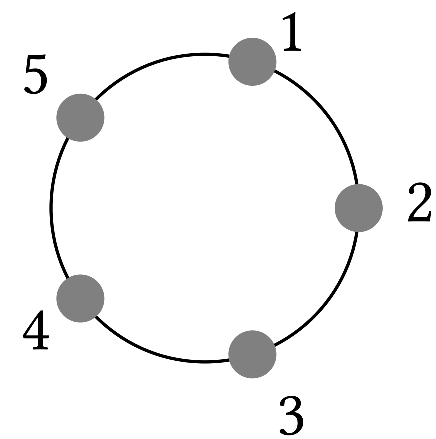


Warm-up: C_5

- the smallest non-complete core
- can you beat 5^{tw}?

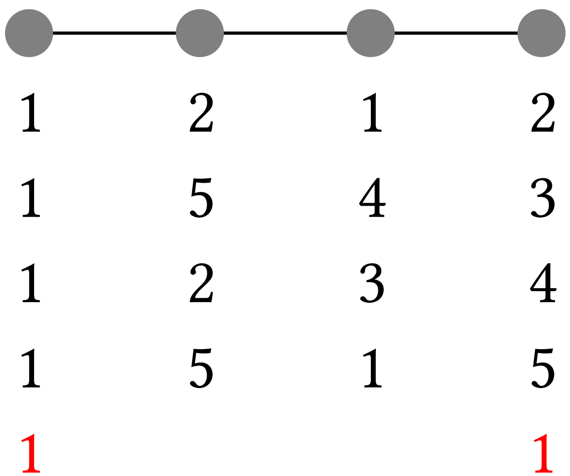


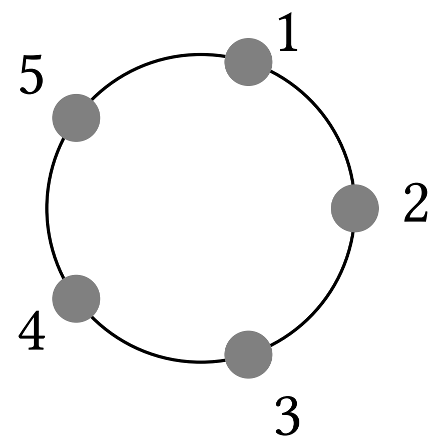
1	2	1	2
1	5	4	3
1	2	3	4
1	5	1	5



Warm-up: C_5

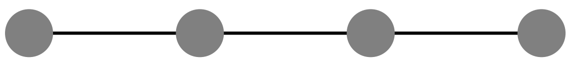
- the smallest non-complete core
- can you beat 5^{tw}?



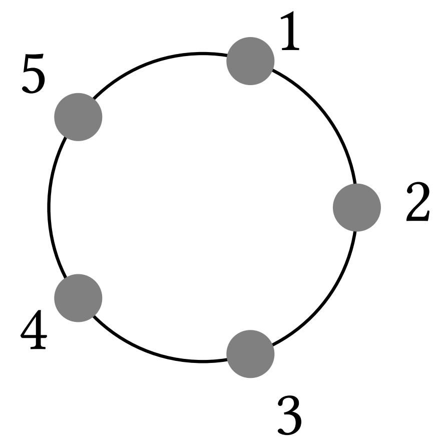


Warm-up: C_5

- the smallest non-complete core
- can you beat 5^{tw}?



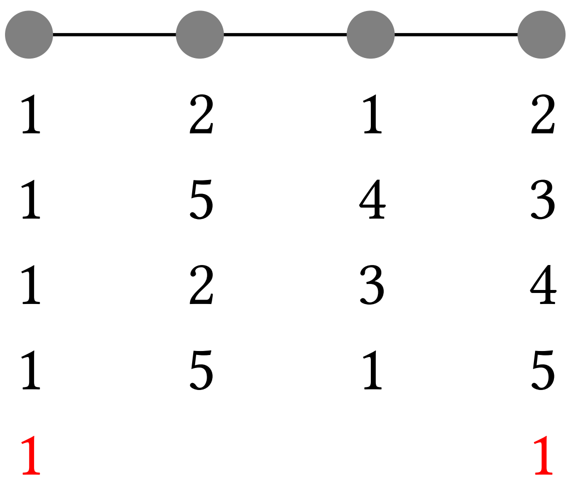
1	2	1	2
1	5	4	3
1	2	3	4
1	5	1	5
1			1

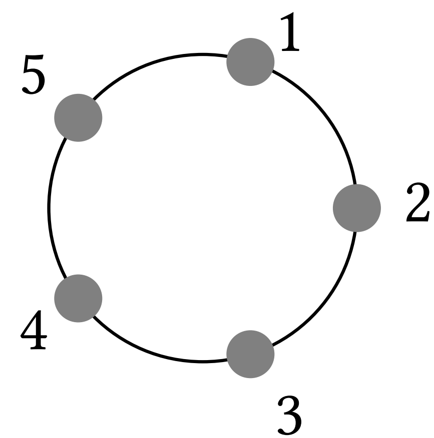


- start with a graph G
- obtain G^* by subdividing each edge twice
- G has a 5-coloring $\Leftrightarrow$ G^* has a C_5-coloring

Warm-up: C_5

- the smallest non-complete core
- can you beat 5^{tw}?

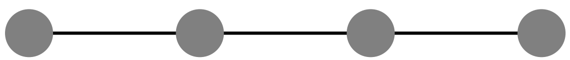




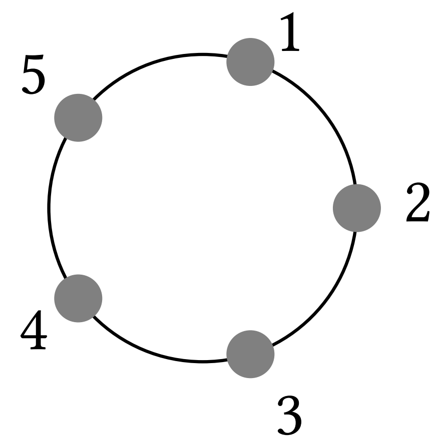
- start with a graph G
- obtain G^* by subdividing each edge twice
- G has a 5-coloring $\Leftrightarrow$ G^* has a C_5-coloring
- $\mathrm{tw}(G^*) = \mathrm{tw}(G)$

Warm-up: C_5

- the smallest non-complete core
- can you beat 5^{tw}?



1	2	1	2
1	5	4	3
1	2	3	4
1	5	1	5
1			1



- start with a graph G
- obtain G^* by subdividing each edge twice
- G has a 5-coloring $\Leftrightarrow$ G^* has a C_5-coloring
- $\mathrm{tw}(G^*) = \mathrm{tw}(G)$
- finding C_5-coloring of G^* in time $(5-\varepsilon)^{\mathrm{tw}(G^*)}$ $\rightarrow$ finding 5-coloring of G in time $(5-\varepsilon)^{\mathrm{tw}(G)}$ $\rightarrow$ the SETH fails

Algorithmic idea: direct products

- for graphs H_1, H_2, we define their direct product $H_1 \times H_2$ as follows:
  - $V(H_1 \times H_2) = V(H_1) \times V(H_2)$
  - $(u_1, u_2)(v_1, v_2) \in E(H_1 \times H_2)$ iff $u_1 v_1 \in E(H_1)$ and $u_2 v_2 \in E(H_2)$

Algorithmic idea: direct products

- for graphs H_1, H_2, we define their direct product $H_1 \times H_2$ as follows:
  - $V(H_1 \times H_2) = V(H_1) \times V(H_2)$
  - $(u_1, u_2)(v_1, v_2) \in E(H_1 \times H_2)$ iff $u_1 v_1 \in E(H_1)$ and $u_2 v_2 \in E(H_2)$

- the generalization to $H_1 \times H_2 \times \ldots \times H_m$ is natural ($\times$ is associative and commutative)

Algorithmic idea: direct products

- for graphs H_1, H_2, we define their direct product $H_1 \times H_2$ as follows:
  - $V(H_1 \times H_2) = V(H_1) \times V(H_2)$
  - $(u_1, u_2)(v_1, v_2) \in E(H_1 \times H_2)$ iff $u_1 v_1 \in E(H_1)$ and $u_2 v_2 \in E(H_2)$

- the generalization to $H_1 \times H_2 \times \ldots \times H_m$ is natural ($\times$ is associative and commutative)

- $G \to H_1 \times H_2 \times \ldots \times H_m$ iff $G \to H_i$ for every $i \in [m]$

Corollary.
If $H = H_1 \times H_2 \times \ldots \times H_k$, where $|H_1| \geqslant |H_2| \geqslant \ldots \geqslant |H_m|$, then H-coloring of G can be solved in time $|H_1|^{tw}$.

Special case: projective graphs

- for any k we have $H^k \to H$, e.g., projections on each coordinate
- H is projective if for all $k \geqslant 2$ projections are the only homomorphisms from H^k to H (up to automorphisms)
- there are non-projective graphs, e.g., all direct products

Special case: projective graphs

- for any k we have $H^k \to H$, e.g., projections on each coordinate
- H is projective if for all $k \geqslant 2$ projections are the only homomorphisms from H^k to H (up to automorphisms)
- there are non-projective graphs, e.g., all direct products

Lemma.

Let H be a projective core. Then there exists an edge gadget, i.e., a graph F with two special vertices x, y, such that:

- for any $h \colon F \to H$ we have $h(x) \neq h(y)$,
- for any distinct $u, v \in V(H)$ there is $h \colon F \to H$, such that $h(x) = u$ and $h(y) = v$.

Construction of the edge gadget

- $V(H) = \{v_1, v_2, \ldots, v_k\}$
- $F = H^{k(k-1)}$

$$x = (\overbrace{x_1, x_1, \ldots, x_1}^{k-1}, \quad \overbrace{x_2, x_2, \ldots, x_2}^{k-1}, \quad \ldots \quad \overbrace{x_k, x_k, \ldots, x_k}^{k-1})$$

$$y = (x_2, x_3, \ldots, x_k, \quad x_1, x_3, \ldots, x_k, \quad \ldots \quad x_1, x_2, \ldots, x_{k-1})$$

Construction of the edge gadget

- $V(H) = \{v_1, v_2, \ldots, v_k\}$
- $F = H^{k(k-1)}$

$$x = (\overbrace{x_1, x_1, \ldots, x_1}^{k-1}, \quad \overbrace{x_2, x_2, \ldots, x_2}^{k-1}, \quad \ldots \quad \overbrace{x_k, x_k, \ldots, x_k}^{k-1})$$

$$y = (x_2, x_3, \ldots, x_k, \quad x_1, x_3, \ldots, x_k, \quad \ldots \quad x_1, x_2, \ldots, x_{k-1})$$

- H is projective: all homomorphisms from F to H are projections $\rightarrow$ there is no homomorphism that maps x and y on the same vertex

Construction of the edge gadget

- $V(H) = \{v_1, v_2, \ldots, v_k\}$
- $F = H^{k(k-1)}$

$$x = (\overbrace{x_1, x_1, \ldots, x_1}^{k-1}, \quad \overbrace{x_2, x_2, \ldots, x_2}^{k-1}, \quad \ldots \quad \overbrace{x_k, x_k, \ldots, x_k}^{k-1})$$
$$y = (x_2, x_3, \ldots, x_k, \quad x_1, x_3, \ldots, x_k, \quad \ldots \quad x_1, x_2, \ldots, x_{k-1})$$

- H is projective: all homomorphisms from F to H are projections $\rightarrow$ there is no homomorphism that maps x and y on the same vertex
- for every distinct u, v there is a coordinate ℓ, such that $x[\ell] = u$ and $y[\ell] = v$ $\rightarrow$ the projection on the ℓ-th corrdinate is a homomorphism that maps x to u and y to v

Projective graphs: lower bound

Lemma.
Let H be a projective core. Then there exists an edge gadget, i.e., a graph F with two special vertices x, y, such that:

- for any $h\colon F \to H$ we have $h(x) \neq h(y)$,
- for any distinct $u, v \in V(H)$ there is $h\colon F \to H$, such that $h(x) = u$ and $h(y) = v$.

Projective graphs: lower bound

Lemma.
Let H be a projective core. Then there exists an edge gadget, i.e., a graph F with two special vertices x, y, such that:

- for any $h \colon F \to H$ we have $h(x) \neq h(y)$,
- for any distinct $u, v \in V(H)$ there is $h \colon F \to H$, such that $h(x) = u$ and $h(y) = v$.

- let G be an instance of k-coloring with $k := |H|$
- construct G^* by replacing every edge with a copy of F
- G is $|H|$-colorable iff G^* is H-colorable

Projective graphs: lower bound

Lemma.

Let H be a projective core. Then there exists an edge gadget, i.e., a graph F with two special vertices x, y, such that:

- for any $h\colon F \to H$ we have $h(x) \neq h(y)$,
- for any distinct $u, v \in V(H)$ there is $h\colon F \to H$, such that $h(x) = u$ and $h(y) = v$.

- let G be an instance of k-coloring with $k := |H|$
- construct G^* by replacing every edge with a copy of F
- G is $|H|$-colorable iff G^* is H-colorable
- $\mathsf{tw}(G^*) \leqslant \mathsf{tw}(G) + |F| = \mathsf{tw}(G) + \mathcal{O}(1)$

Projective graphs: lower bound

Lemma.
Let H be a projective core. Then there exists an edge gadget, i.e., a graph F with two special vertices x, y, such that:

- for any $h\colon F \to H$ we have $h(x) \neq h(y)$,
- for any distinct $u, v \in V(H)$ there is $h\colon F \to H$, such that $h(x) = u$ and $h(y) = v$.

- let G be an instance of k-coloring with $k := |H|$
- construct G^* by replacing every edge with a copy of F
- G is $|H|$-colorable iff G^* is H-colorable
- $\mathsf{tw}(G^*) \leqslant \mathsf{tw}(G) + |F| = \mathsf{tw}(G) + \mathcal{O}(1)$
- finding H-coloring of G^* in time $(k-\varepsilon)^{\mathsf{tw}(G^*)} \to$ finding k-coloring of G in time $(k-\varepsilon)^{\mathsf{tw}(G)} \to$ the SETH fails

Projective vs. non-projective graphs

Theorem (Okrasa, Rz.)

Let H be a projective core. Then H-coloring of G cannot be solved in time $(|H| - \varepsilon)^{\mathrm{tw}}$, unless the SETH fails.

- tight: the straightforward algorithm works in time $|H|^{\mathrm{tw}(G)}$

Projective vs. non-projective graphs

Theorem (Okrasa, Rz.)

Let H be a projective core. Then H-coloring of G cannot be solved in time $(|H| - \varepsilon)^{tw}$, unless the SETH fails.

- tight: the straightforward algorithm works in time $|H|^{tw(G)}$

Theorem (Hell, Nešetřil + Łuczak, Nešetřil).

Almost all graphs are projective cores.

Projective vs. non-projective graphs

Theorem (Okrasa, Rz.)

Let H be a projective core. Then H-coloring of G cannot be solved in time $(|H| - \varepsilon)^{tw}$, unless the SETH fails.

- tight: the straightforward algorithm works in time $|H|^{tw(G)}$

Theorem (Hell, Nešetřil + Łuczak, Nešetřil).

Almost all graphs are projective cores.

- can we do the same for non-projective graphs?

Projective vs. non-projective graphs

Theorem (Okrasa, Rz.)
Let H be a projective core. Then H-coloring of G cannot be solved in time $(|H| - \varepsilon)^{tw}$, unless the SETH fails.

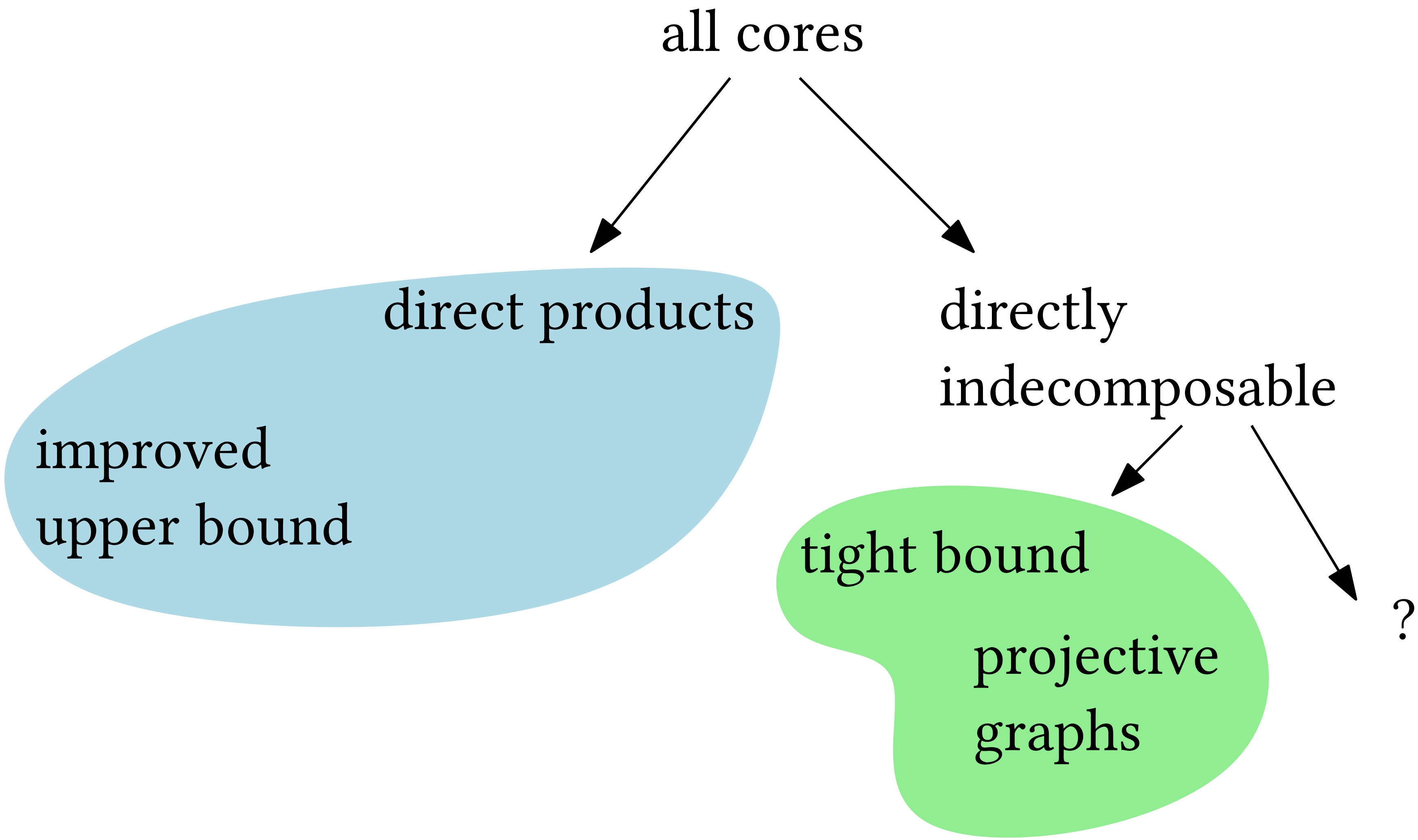
- tight: the straightforward algorithm works in time $|H|^{tw(G)}$

Theorem (Hell, Nešetřil + Łuczak, Nešetřil).
Almost all graphs are projective cores.

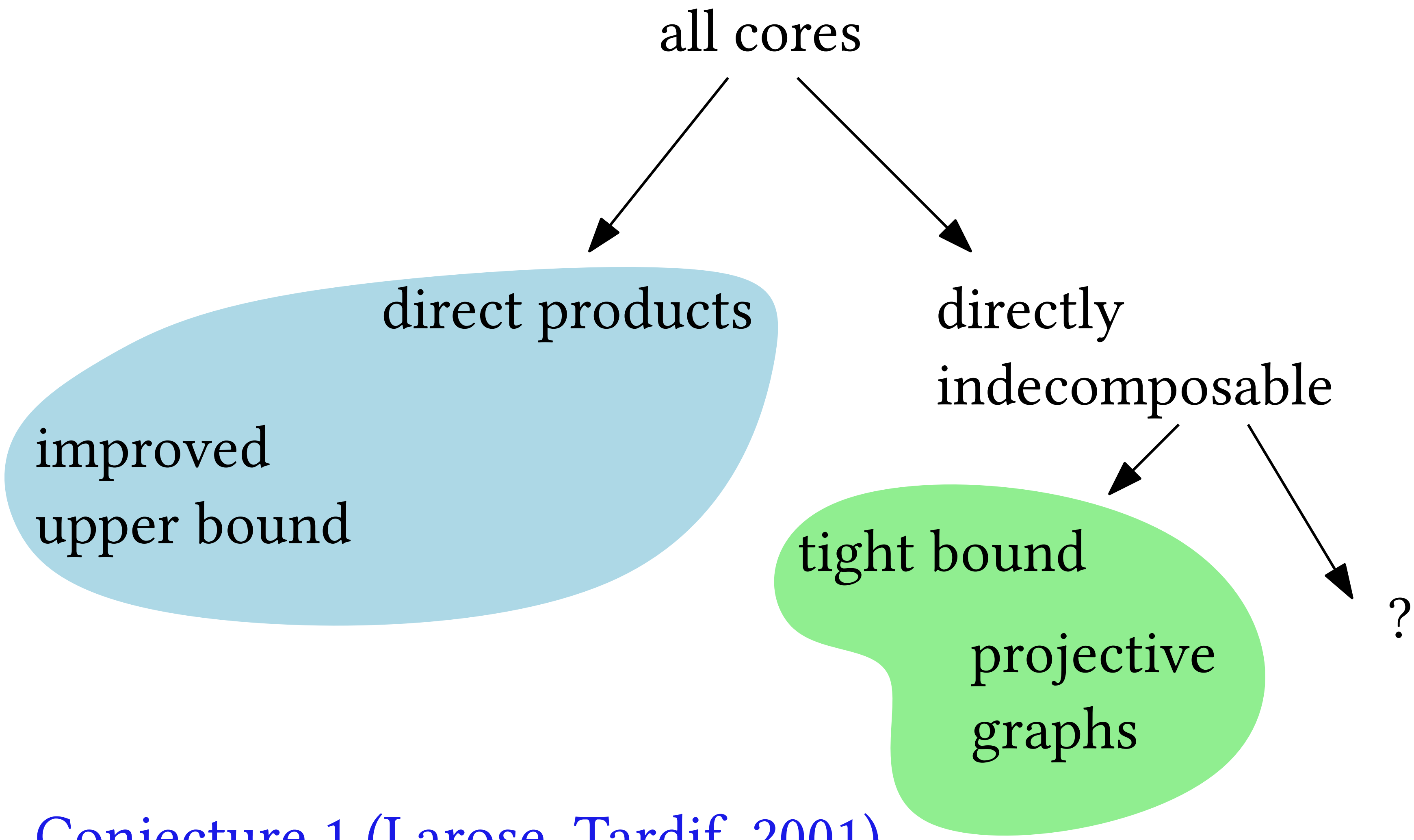
- can we do the same for non-projective graphs?

Proposition.
There exists an edge gadget for H if and only if H is projective.

Overview on the current situation



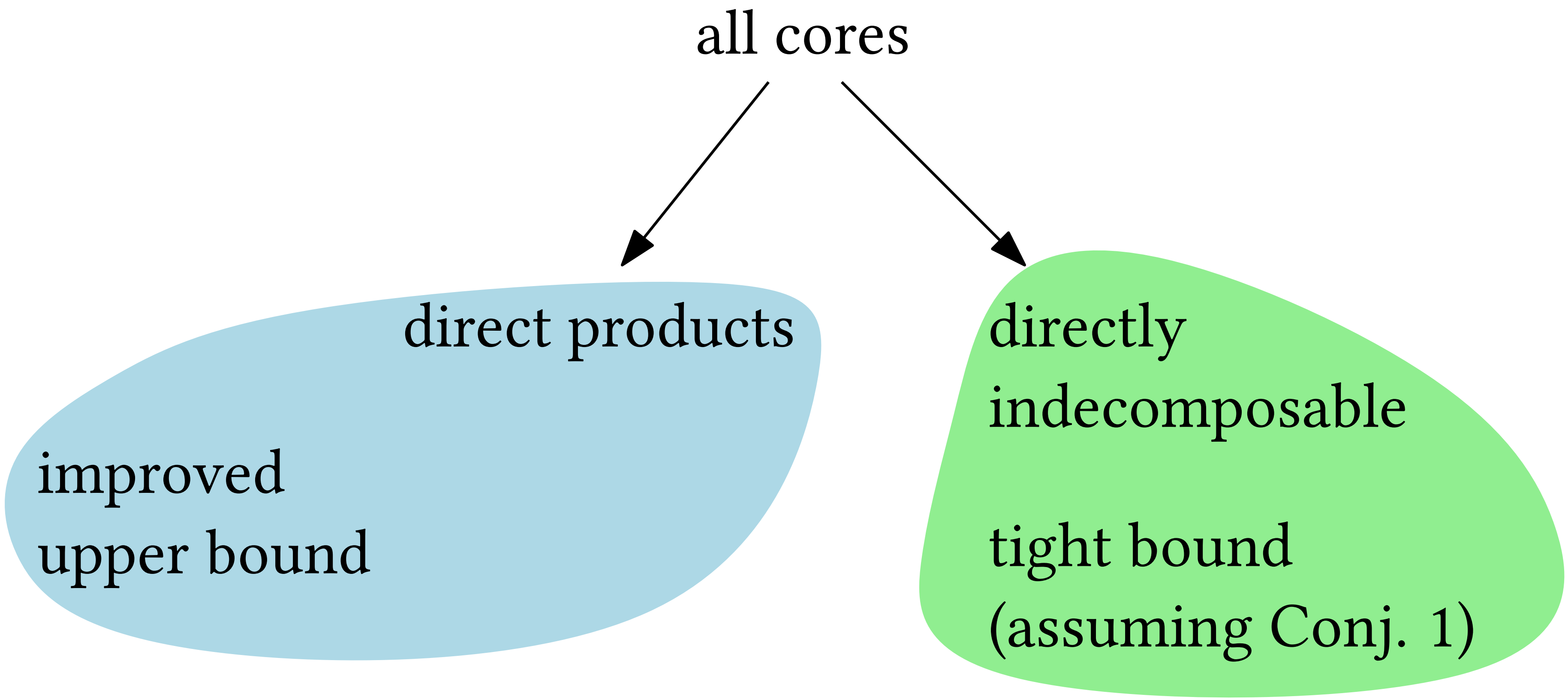
Overview on the current situation



Conjecture 1 (Larose, Tardif, 2001).

A connected non-bipartite core is indecomposable iff it is projective.

Overview on the current situation



Conjecture 1 (Larose, Tardif, 2001).

A connected non-bipartite core is indecomposable iff it is projective.

Direct products of graphs – a closer look

- consider $H = H_1 \times H_2 \times \ldots \times H_m$, where $|H_1| \geqslant \ldots \geqslant |H_m|$ and each H_i is indecomposable
- since H is a core, each H_i is a core
- Conjecture 1 implies that each H_i is projective
- recall that we can solve H-coloring of G in time $|H_1|^{\mathsf{tw}(G)}$

Direct products of graphs – a closer look

- consider $H = H_1 \times H_2 \times \ldots \times H_m$,
  where $|H_1| \geqslant \ldots \geqslant |H_m|$ and each H_i is indecomposable
- since H is a core, each H_i is a core
- Conjecture 1 implies that each H_i is projective
- recall that we can solve H-coloring of G in time $|H_1|^{\mathsf{tw}(G)}$
- Larose (2002) studied a subclass of projective graphs, called strongly projective

Theorem (Okrasa, Rz.).
If H_1 is strongly projective, then H-coloring of G cannot be solved in time $(|H_1| - \varepsilon)^{\mathsf{tw}(G)}$, unless the SETH fails.

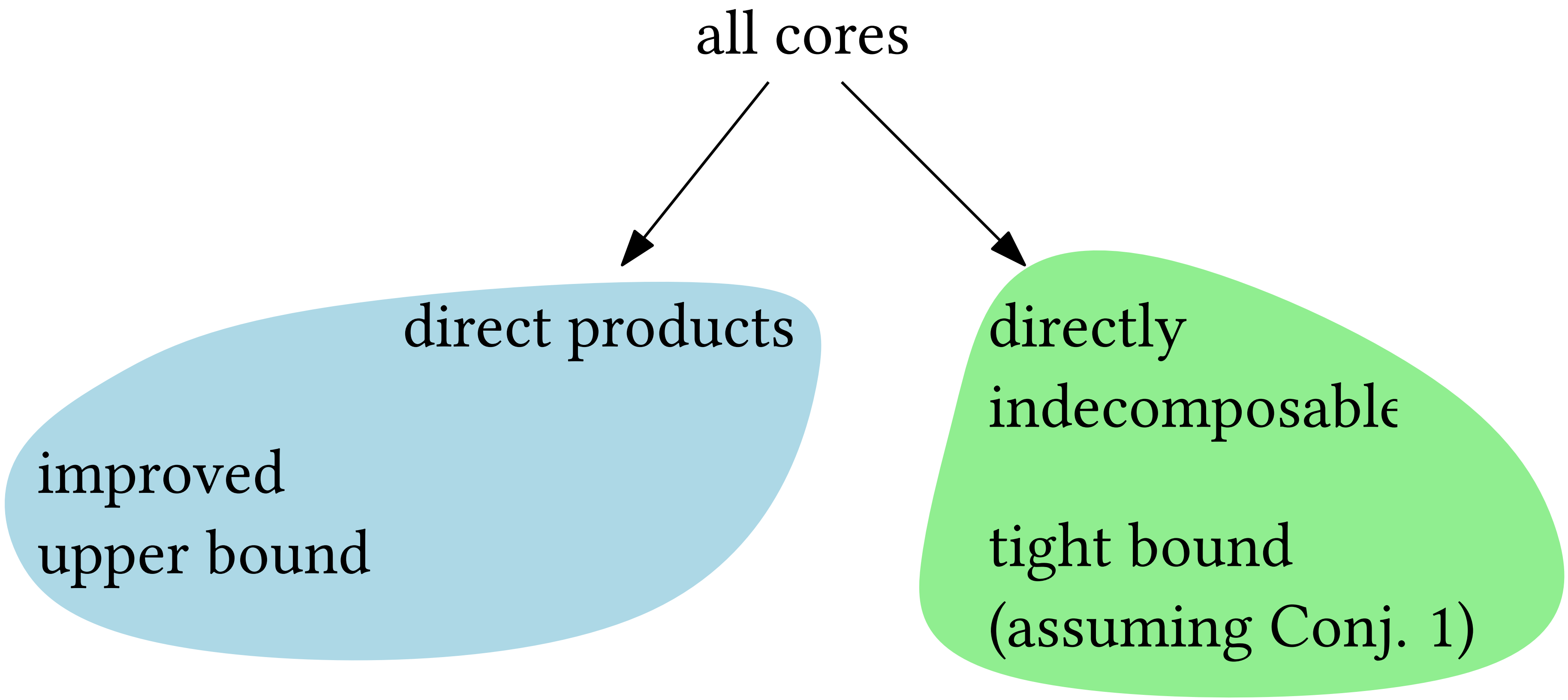
Direct products of graphs – a closer look

- consider $H = H_1 \times H_2 \times \ldots \times H_m$,
  where $|H_1| \geqslant \ldots \geqslant |H_m|$ and each H_i is indecomposable
- since H is a core, each H_i is a core
- Conjecture 1 implies that each H_i is projective
- recall that we can solve H-coloring of G in time $|H_1|^{\mathsf{tw}(G)}$
- Larose (2002) studied a subclass of projective graphs, called strongly projective

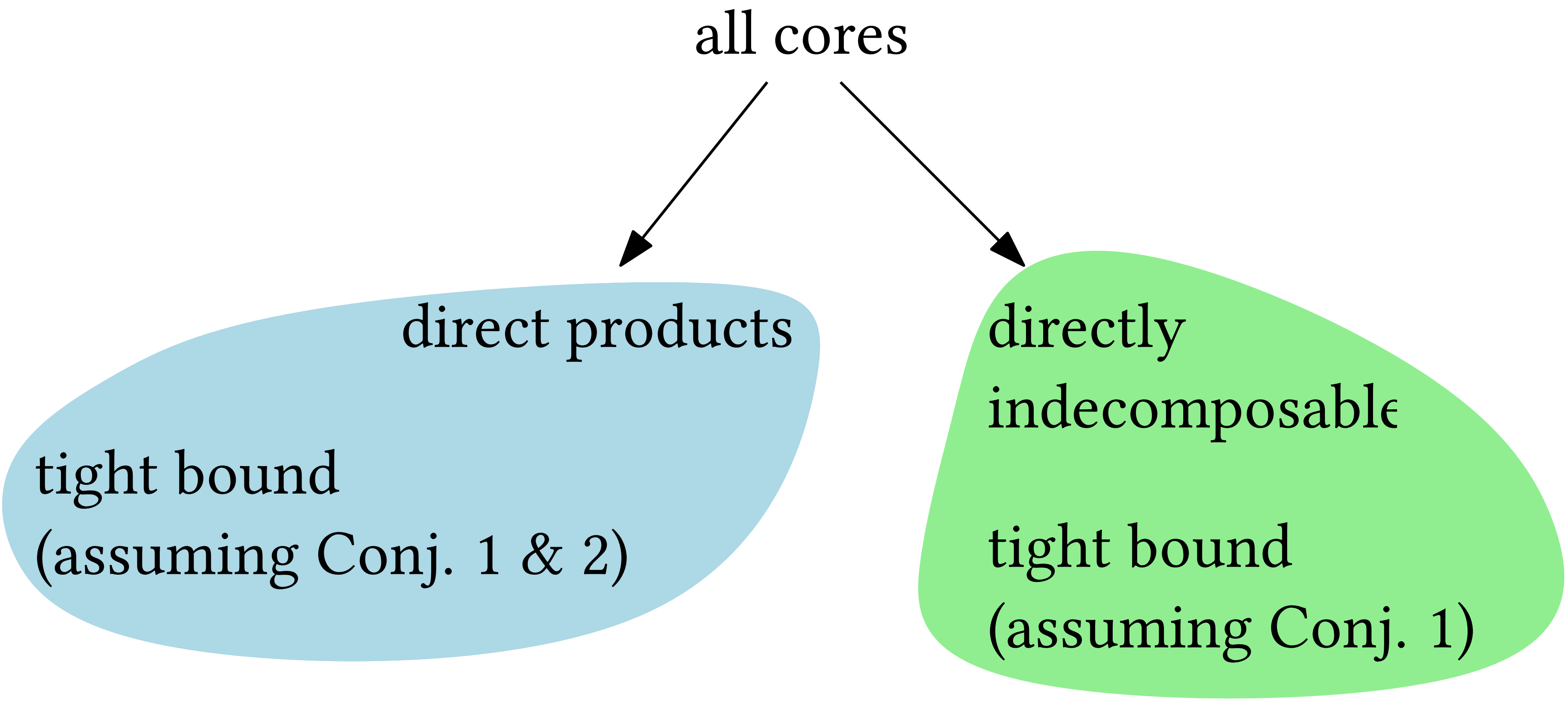
Theorem (Okrasa, Rz.).
If H_1 is strongly projective, then H-coloring of G cannot be solved in time $(|H_1| - \varepsilon)^{\mathsf{tw}(G)}$, unless the SETH fails.

Conjecture 2 (Larose, 2002).
Every projective core is strongly projective.

Final overview



Final overview



Theorem (Okrasa, Rz.).

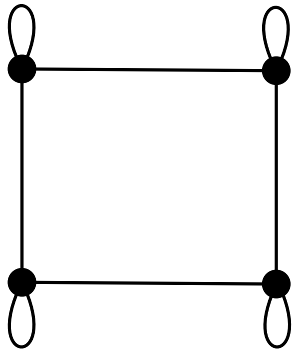
Assume Conjectures 1 and 2. Let $H = H_1 \times \ldots \times H_m$ be a core, where $|H_1| \geqslant \ldots \geqslant |H_m|$. Then H-coloring of G

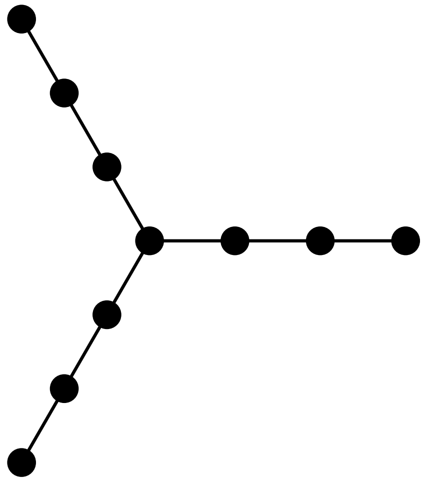
a) can be solved in time $|H_1|^{\mathsf{tw}(G)}$,

b) cannot be solved in time $(|H_1| - \varepsilon)^{\mathsf{tw}(G)}$, under the SETH.

List homomorphisms

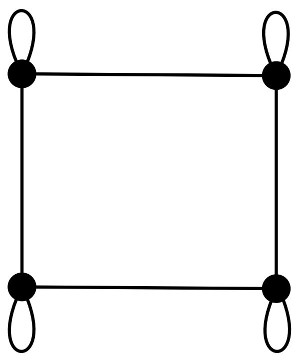
- each vertex v of G has a list $L(v)$ of vertices of H
- v can only be mapped to a vertex from $L(v)$
- it's a *harder* problem, in particular has more NP-hard cases

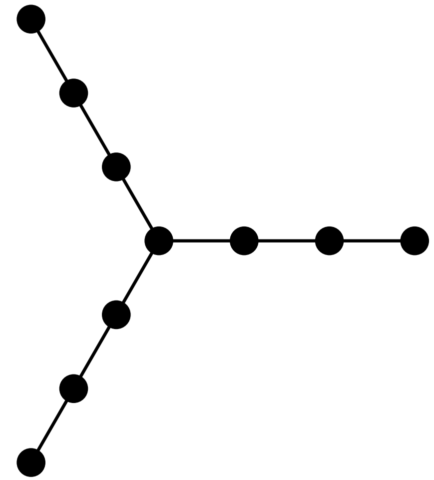




List homomorphisms

- each vertex v of G has a list $L(v)$ of vertices of H
- v can only be mapped to a vertex from $L(v)$
- it's a *harder* problem, in particular has more NP-hard cases





Problem.

For every graph H, find $k = k(H)$, such that list H-coloring of G

- can be solved in time k^{tw},
- cannot be solved in time $(k - \varepsilon)^{tw}$, unless the SETH fails.

- we still have $k \leqslant |H|$ and $k = |H|$ if H is complete

Algorithmic idea: incomparable vertices

- actually, we have

$$k \leqslant \max_{v \in V(G)} |L(v)|.$$

this is not a property of H

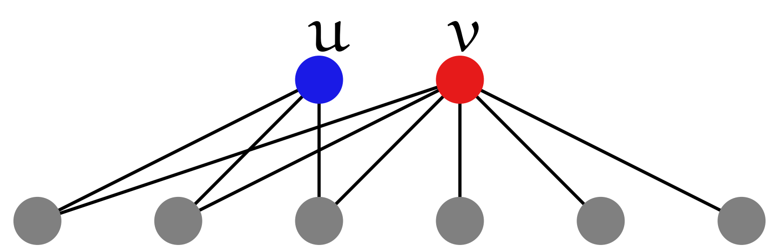
Algorithmic idea: incomparable vertices

- actually, we have

$$k \leqslant \max_{v \in V(G)} |L(v)|.$$

this is not a property of H ... or is it?

- $u, v \in V(H)$ are comparable if $N(u) \subseteq N(v)$



- if both appear in one list, we can safely remove u
- no list contains two comparable vertices

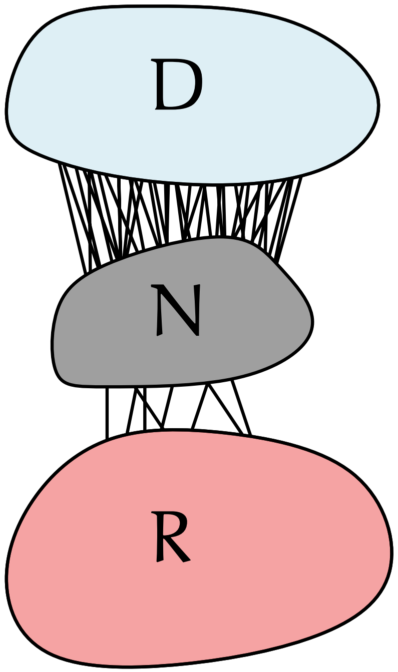
$i(H)$ = size of the largest set of pairwise incomparable vertices

- $k \leqslant i(H)$

One more algorithmic idea: decomposition

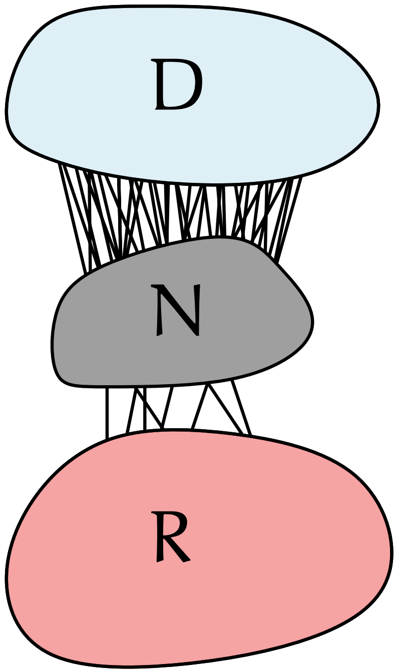
- we found three types of decompositions of H

- N separates D and R
- N is a reflexive clique
- all edges between D and N are in H



One more algorithmic idea: decomposition

- we found three types of decompositions of H



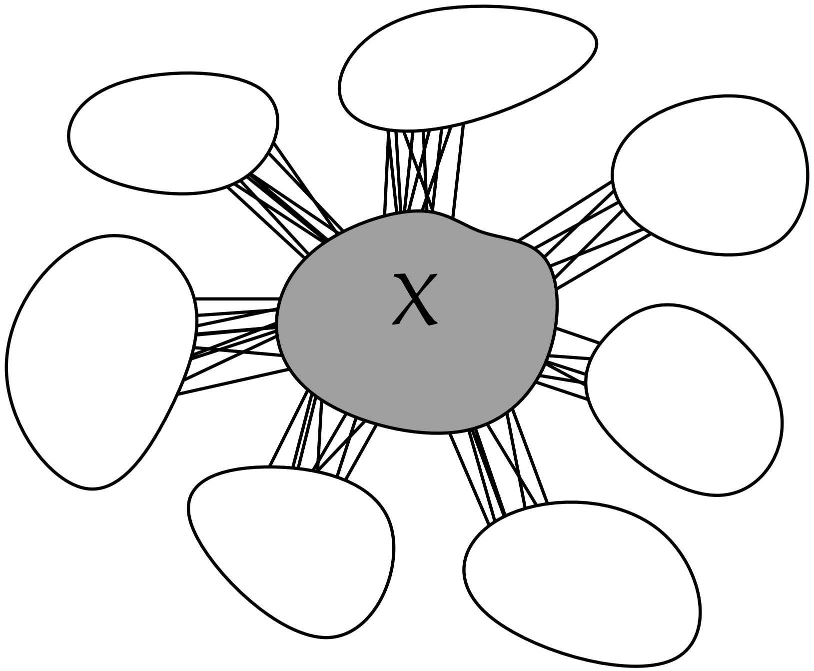
- N separates D and R
- N is a reflexive clique
- all edges between D and N are in H

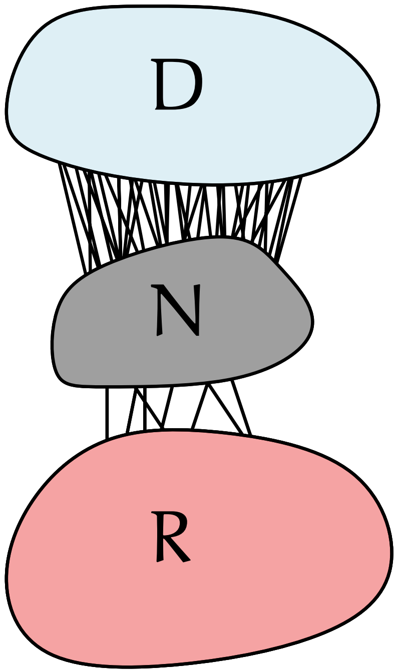
- for $u \in D$ and $v \in N$, always $N[u] \subseteq N[v]$
- vertices from D and N never appear in the same list

- if $x \in V(G)$ is mapped to D and $y \in V(G)$ is mapped to R, then every x-y path contains a vertex mapped to N

Decomposition lemma

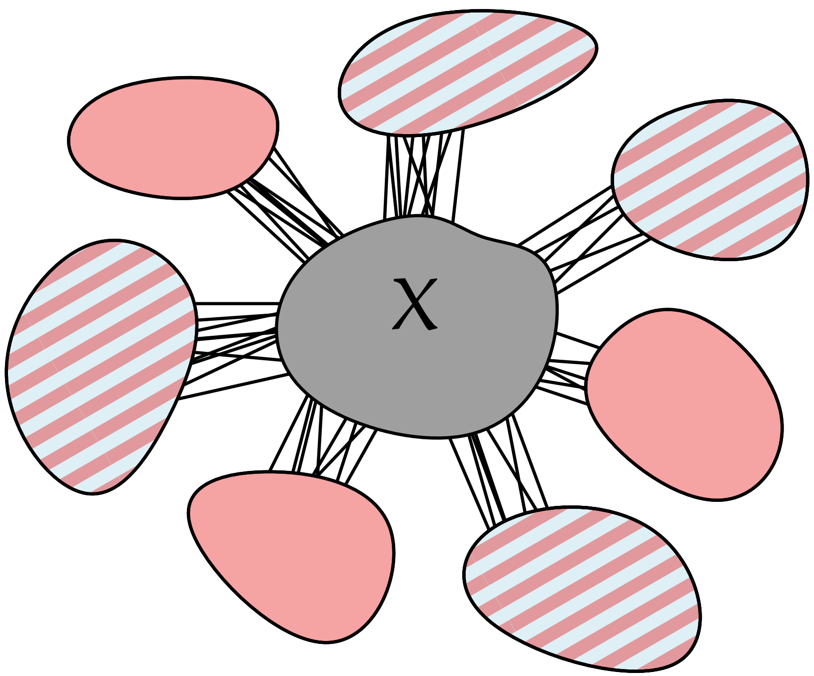
$X = \{v \colon L(v) \cap N \neq \emptyset\}$, $\mathcal{C}$ = set of components of $G - X$

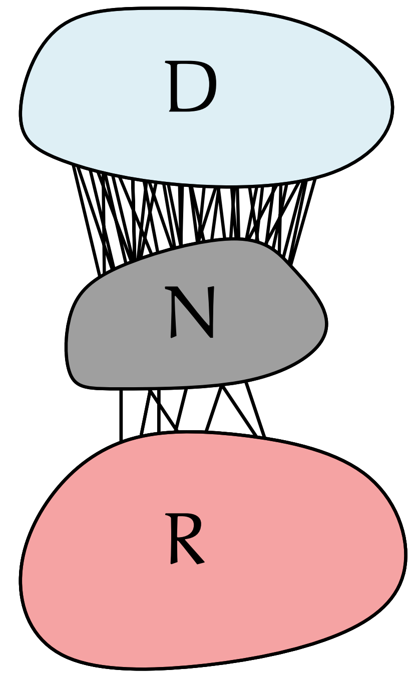




Decomposition lemma

$X = \{v \colon L(v) \cap N \neq \emptyset\}$, $\mathcal{C}$ = set of components of $G - X$

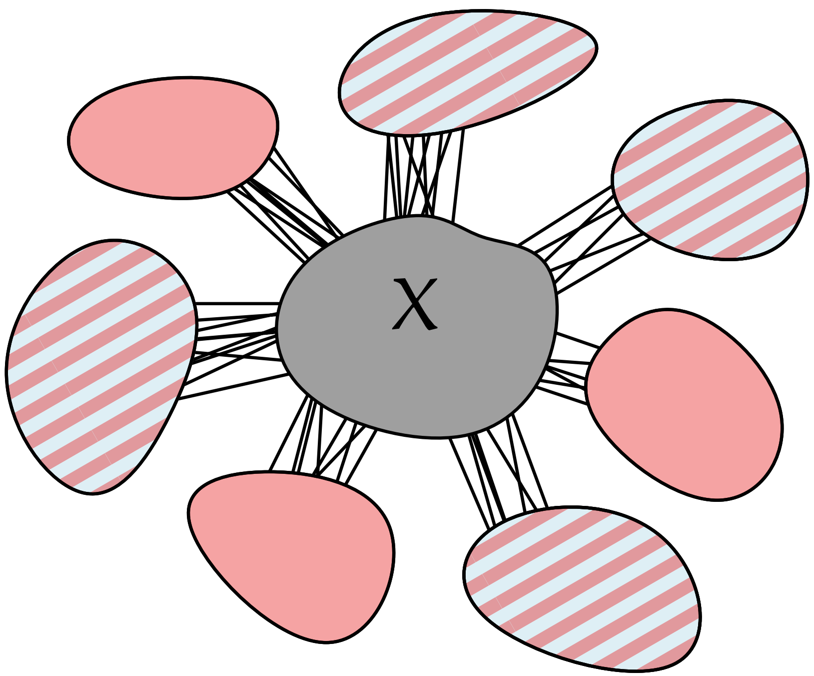


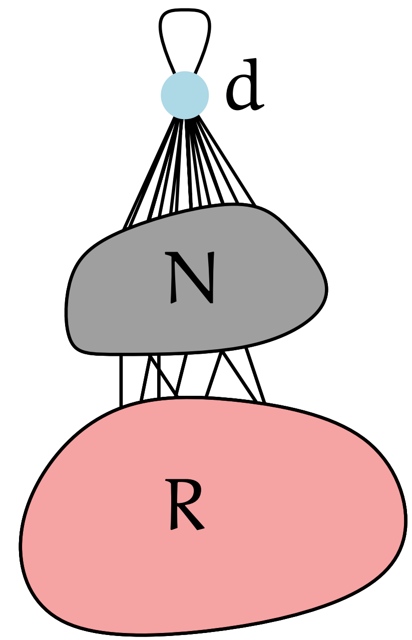


- every $C \in \mathcal{C}$ must be entirely mapped to D or R
- precompute list $H[D]$-coloring of each $C \in \mathcal{C}$

Decomposition lemma

$X = \{v\colon L(v) \cap N \neq \emptyset\}$, $\mathcal{C}$ = set of components of $G - X$





- every $C \in \mathcal{C}$ must be entirely mapped to D or R
- precompute list $H[D]$-coloring of each $C \in \mathcal{C}$
- collapse D to a single reflexive vertex d, obtaining H'
- update lists: if v is in $C \in \mathcal{C}$, which can be mapped to D, then add d to $L(v)$

Decomposition lemma, continued

- we reduced an instance of list H-Coloring to $n^{\mathcal{O}(1)}$ instances of list H[D]-Coloring and list H'-Coloring

Decomposition lemma.

If we can solve list H[D]-Coloring and list H'-Coloring in time c^{tw}, then we can solve list H-Coloring in time c^{tw}.

Decomposition lemma, continued

- we reduced an instance of list H-Coloring to $n^{\mathcal{O}(1)}$ instances of list H[D]-Coloring and list H'-Coloring

Decomposition lemma.

If we can solve list H[D]-Coloring and list H'-Coloring in time c^{tw}, then we can solve list H-Coloring in time c^{tw}.

$i^*(H) \approx$ maximum $i(H')$ over all undecomposable induced subgraphs of H

Decomposition lemma, continued

- we reduced an instance of list H-Coloring to $n^{\mathcal{O}(1)}$ instances of list $H[D]$-Coloring and list H'-Coloring

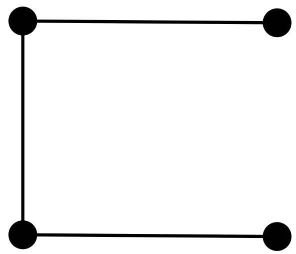
Decomposition lemma.
If we can solve list $H[D]$-Coloring and list H'-Coloring in time c^{tw}, then we can solve list H-Coloring in time c^{tw}.

$i^*(H) \approx$ maximum $i(H')$ over all undecomposable induced subgraphs of H

Theorem (Egri, Marx, Rz. + Okrasa, Piecyk, Rz.).
For every graph “hard” H, $i^*(H)$ is the correct bound:
a) List H-Coloring can be solved in time $(i^*(H))^{\mathrm{tw}}$,
b) List H-Coloring cannot be solved in time $(i^*(H) - \varepsilon)^{\mathrm{tw}}$ (SETH).

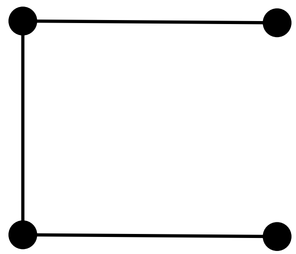
Counting list homomorpshisms

- what is the number of list H-colorings?
- even more hard cases than for the decision variant

counting  colorings ≡ counting independent sets in bipartite graphs

Counting list homomorpshisms

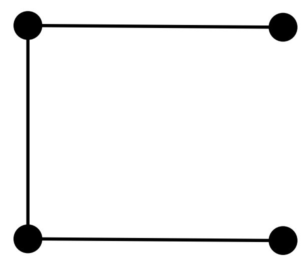
- what is the number of list H-colorings?
- even more hard cases than for the decision variant

counting  colorings $\equiv$ counting independent sets in bipartite graphs

- incomparable vertices do not work this time
- but we can assume each list is irredundant – has no vertices with exactly the same neighborhood $\rightarrow |L(v)| \leqslant \mathsf{irr}(H)$

Counting list homomorpshisms

- what is the number of list H-colorings?
- even more hard cases than for the decision variant

counting  colorings $\equiv$ counting independent sets in bipartite graphs

- incomparable vertices do not work this time
- but we can assume each list is irredundant – has no vertices with exactly the same neighborhood $\to |L(v)| \leqslant \mathsf{irr}(H)$

Theorem (Focke, Marx, Rz.).
For every graph "hard" H, $\mathsf{irr}(H)$ is the correct bound:
a) #List H-Coloring can be solved in time $(\mathsf{irr}(H))^{\mathsf{tw}}$,
b) #List H-Coloring cannot be solved in time $(\mathsf{irr}(H) - \varepsilon)^{\mathsf{tw}}$ (#SETH).

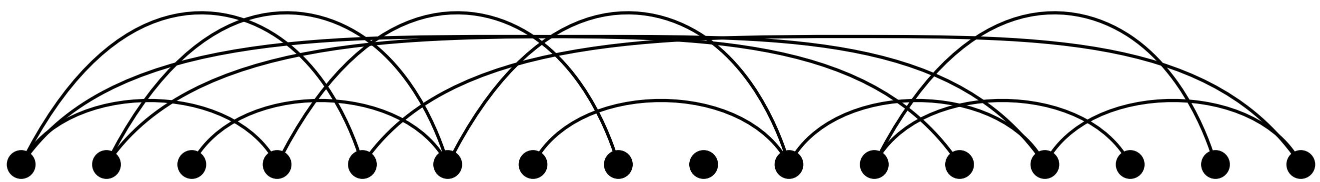
What's next?

- similar results are known for cliquewidth
  [Ganian, Hamm, Korchemna, Okrasa, Simonov]

What's next?

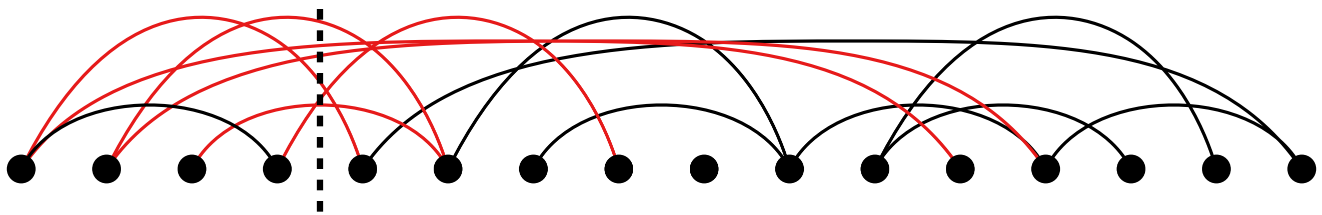
- similar results are known for cliquewidth
  [Ganian, Hamm, Korchemna, Okrasa, Simonov]

- cutwidth – minimize the number of edges crossed by a cut, over all vertex orderings



What's next?

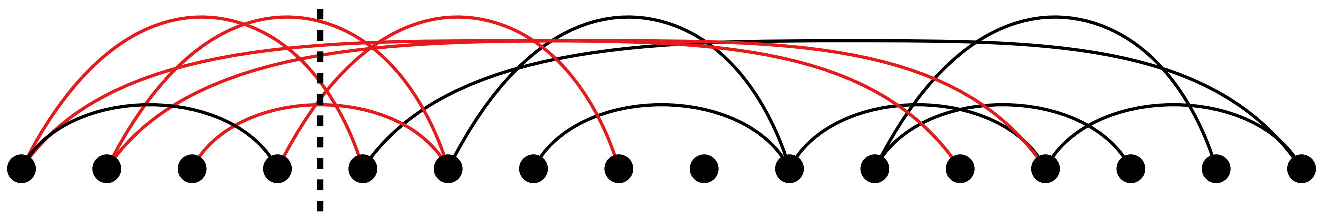
- similar results are known for cliquewidth
  [Ganian, Hamm, Korchemna, Okrasa, Simonov]
- cutwidth – minimize the number of edges crossed by a cut, over all vertex orderings



What's next?

- similar results are known for cliquewidth
  [Ganian, Hamm, Korchemna, Okrasa, Simonov]

- cutwidth – minimize the number of edges crossed by a cut, over all vertex orderings

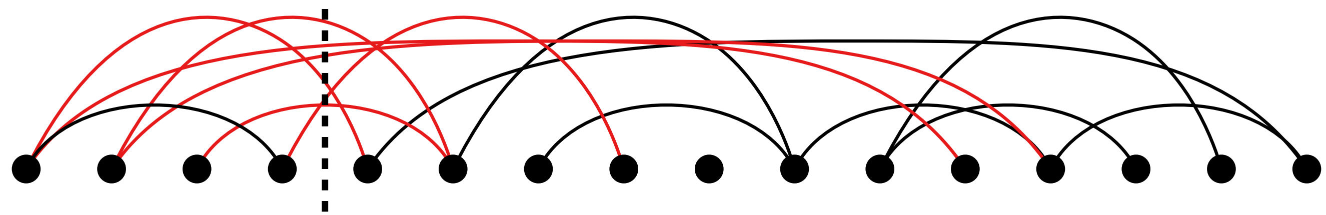


Theorem (Jansen, Nederlof).
(List) k-Coloring can be solved in (randomized) time 2^{ctw}.

What's next?

- similar results are known for cliquewidth
  [Ganian, Hamm, Korchemna, Okrasa, Simonov]

- cutwidth – minimize the number of edges crossed by a cut, over all vertex orderings



Theorem (Jansen, Nederlof).
(List) k-Coloring can be solved in (randomized) time 2^{ctw}.

- complexity of CSP parameterized by the structure of the Gaifman graph

Even if meant for kids, still fun for adults