Sparsity

tutorial at PCC'20

Michał Pilipczuk

Faculty of Mathematics, Informatics, and Mechanics
University of Warsaw

September 17th, 2020

Organization

Organization

Rough plan:

9:15 – 10:00	**Introduction**
10:15 – 11:00	**Generalized coloring numbers**
11:15 – 12:00	**Treedepth and low treedepth colorings**
12:15 – 13:00	**Uniform quasi-wideness and ladders**

Organization

Rough plan:

9:15 – 10:00	**Introduction**
10:15 – 11:00	**Generalized coloring numbers**
11:15 – 12:00	**Treedepth and low treedepth colorings**
12:15 – 13:00	**Uniform quasi-wideness and ladders**

Format:

– Lecture interleaved with short exercises. $\rightsquigarrow$ **Be active!**

– Understanding checks by writing **+1** in the chat.

Sparsity

Sparsity

Graphs in applications are often **sparse**.

Sparsity

Graphs in applications are often **sparse**.

– Transportation networks are (roughly) planar.

Sparsity

Graphs in applications are often **sparse**.

- Transportation networks are (roughly) planar.
- Facebook graph has average degree 338 and median degree 200.

Sparsity

Graphs in applications are often **sparse**.

- Transportation networks are (roughly) planar.
- Facebook graph has average degree 338 and median degree 200.

What does it mean **sparse**?

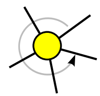
Sparsity

Graphs in applications are often **sparse**.

- Transportation networks are (roughly) planar.
- Facebook graph has average degree 338 and median degree 200.

What does it mean **sparse**?

- Bounded degree?



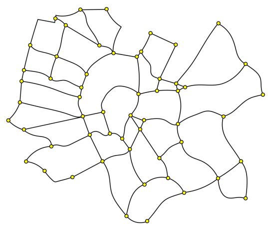
Sparsity

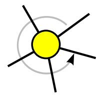
Graphs in applications are often **sparse**.

- Transportation networks are (roughly) planar.
- Facebook graph has average degree 338 and median degree 200.

What does it mean **sparse**?

- Bounded degree?
- Planar-like? Tree-like?





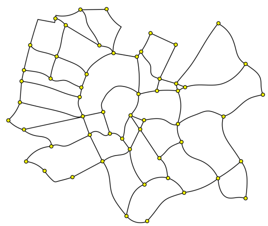
Sparsity

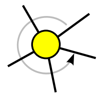
Graphs in applications are often **sparse**.

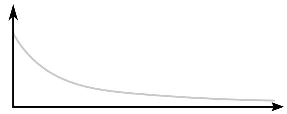
- Transportation networks are (roughly) planar.
- Facebook graph has average degree 338 and median degree 200.

What does it mean **sparse**?

- Bounded degree?
- Planar-like? Tree-like?
- Fixed degree distribution?







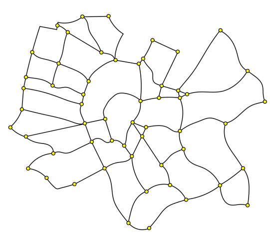
Sparsity

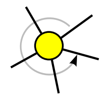
Graphs in applications are often **sparse**.

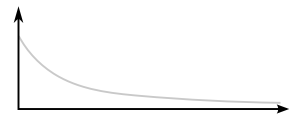
- Transportation networks are (roughly) planar.
- Facebook graph has average degree 338 and median degree 200.

What does it mean **sparse**?

- Bounded degree?
- Planar-like? Tree-like?
- Fixed degree distribution?







Goal. A **mathematical theory** of sparse graphs that is:

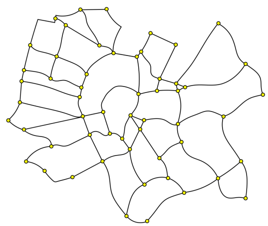
Sparsity

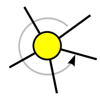
Graphs in applications are often **sparse**.

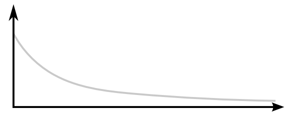
- Transportation networks are (roughly) planar.
- Facebook graph has average degree 338 and median degree 200.

What does it mean **sparse**?

- Bounded degree?
- Planar-like? Tree-like?
- Fixed degree distribution?







Goal. A **mathematical theory** of sparse graphs that is:

1. general and robust;

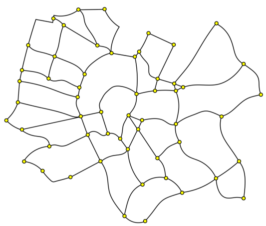
Sparsity

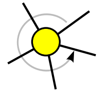
Graphs in applications are often **sparse**.

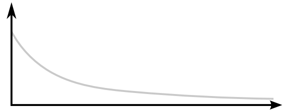
- Transportation networks are (roughly) planar.
- Facebook graph has average degree 338 and median degree 200.

What does it mean **sparse**?

- Bounded degree?
- Planar-like? Tree-like?
- Fixed degree distribution?







Goal. A **mathematical theory** of sparse graphs that is:

1. general and robust;

2. elegant and interesting;

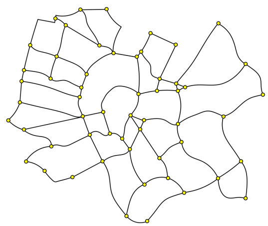
Sparsity

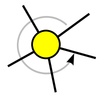
Graphs in applications are often **sparse**.

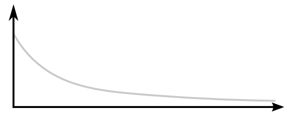
- Transportation networks are (roughly) planar.
- Facebook graph has average degree 338 and median degree 200.

What does it mean **sparse**?

- Bounded degree?
- Planar-like? Tree-like?
- Fixed degree distribution?







Goal. A **mathematical theory** of sparse graphs that is:

1. general and robust;

2. elegant and interesting;

3. useful in applications.

Sparsity

Graphs in applications are often **sparse**.

– Transportation networks are (roughly) planar.

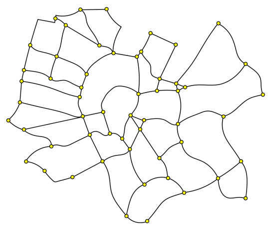
– Facebook graph has average degree 338 and median degree 200.

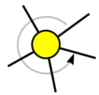
What does it mean **sparse**?

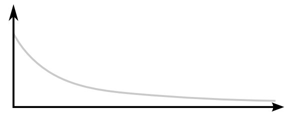
– Bounded degree?

– Planar-like? Tree-like?

– Fixed degree distribution?







Goal. A **mathematical theory** of sparse graphs that is:

1. general and robust;

2. elegant and interesting;

3. useful in applications.

Sparsity: a young area of graph theory that $\pm$ achieves all of the above.

Measuring sparsity

Measuring sparsity

Q: What does it mean that a graph is **sparse**?

Measuring sparsity

Q: What does it mean that a graph is **sparse**?

Attempt 1. A graph G is **sparse** if it has a linear number of edges.

Measuring sparsity

Q: What does it mean that a graph is **sparse**?

Attempt 1. A graph G is **sparse** if it has a linear number of edges.

– Formally, $|E(G)| \leqslant c \cdot |V(G)|$ for some constant c.

Measuring sparsity

Q: What does it mean that a graph is **sparse**?

Attempt 1. A graph G is **sparse** if it has a linear number of edges.

– Formally, $|E(G)| \leqslant c \cdot |V(G)|$ for some constant c.

$$\mathrm{avgdeg}(G) = \frac{\sum_{u \in V(G)} \deg(u)}{|V(G)|} = \frac{2|E(G)|}{|V(G)|}$$

Measuring sparsity

Q: What does it mean that a graph is **sparse**?

Attempt 1. A graph G is **sparse** if it has a linear number of edges.

– Formally, $|E(G)| \leqslant c \cdot |V(G)|$ for some constant c.

$$\mathrm{avgdeg}(G) = \frac{\sum_{u \in V(G)} \deg(u)}{|V(G)|} = \frac{2|E(G)|}{|V(G)|}$$

– Equivalently, average degree in G is bounded by $2c$.

Measuring sparsity

Q: What does it mean that a graph is **sparse**?

Attempt 1. A graph G is **sparse** if it has a linear number of edges.

– Formally, $|E(G)| \leqslant c \cdot |V(G)|$ for some constant c.

$$\mathrm{avgdeg}(G) = \frac{\sum_{u \in V(G)} \deg(u)}{|V(G)|} = \frac{2|E(G)|}{|V(G)|}$$

– Equivalently, average degree in G is bounded by $2c$.

Ex 1. Maximum degree $\leqslant d \quad \Rightarrow \quad$ Average degree $\leqslant d$.

Measuring sparsity

Q: What does it mean that a graph is **sparse**?

Attempt 1. A graph G is **sparse** if it has a linear number of edges.

– Formally, $|E(G)| \leqslant c \cdot |V(G)|$ for some constant c.

$$\mathrm{avgdeg}(G) = \frac{\sum_{u \in V(G)} \deg(u)}{|V(G)|} = \frac{2|E(G)|}{|V(G)|}$$

– Equivalently, average degree in G is bounded by $2c$.

Ex 1. Maximum degree $\leqslant d \quad \Rightarrow \quad$ Average degree $\leqslant d$.

Ex 2. Planar graph has $\leqslant 3n - 6$ edges $\quad \Rightarrow \quad$ Average degree < 6.

Measuring sparsity

Q: What does it mean that a graph is **sparse**?

Attempt 1. A graph G is **sparse** if it has a linear number of edges.

– Formally, $|E(G)| \leqslant c \cdot |V(G)|$ for some constant c.

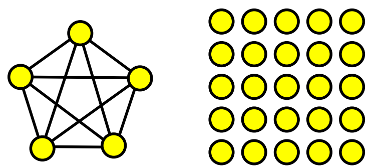
$$\mathrm{avgdeg}(G) = \frac{\sum_{u \in V(G)} \deg(u)}{|V(G)|} = \frac{2|E(G)|}{|V(G)|}$$

– Equivalently, average degree in G is bounded by $2c$.

Ex 1. Maximum degree $\leqslant d \quad \Rightarrow \quad$ Average degree $\leqslant d$.

Ex 2. Planar graph has $\leqslant 3n - 6$ edges $\quad \Rightarrow \quad$ Average degree < 6.

Issue: A complete graph on k vertices plus k^2 isolated vertices.



Measuring sparsity

Q: What does it mean that a graph is **sparse**?

Attempt 1. A graph G is **sparse** if it has a linear number of edges.

– Formally, $|E(G)| \leqslant c \cdot |V(G)|$ for some constant c.

$$\mathrm{avgdeg}(G) = \frac{\sum_{u \in V(G)} \deg(u)}{|V(G)|} = \frac{2|E(G)|}{|V(G)|}$$

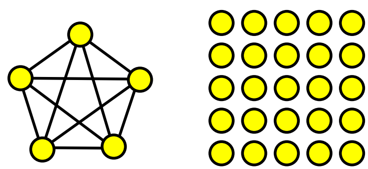
– Equivalently, average degree in G is bounded by $2c$.

Ex 1. Maximum degree $\leqslant d \quad \Rightarrow \quad$ Average degree $\leqslant d$.

Ex 2. Planar graph has $\leqslant 3n - 6$ edges $\quad \Rightarrow \quad$ Average degree < 6.

Issue: A complete graph on k vertices plus k^2 isolated vertices.

– Average degree smaller than 1.



Measuring sparsity

Q: What does it mean that a graph is **sparse**?

Attempt 1. A graph G is **sparse** if it has a linear number of edges.

– Formally, $|E(G)| \leqslant c \cdot |V(G)|$ for some constant c.

$$\mathrm{avgdeg}(G) = \frac{\sum_{u \in V(G)} \deg(u)}{|V(G)|} = \frac{2|E(G)|}{|V(G)|}$$

– Equivalently, average degree in G is bounded by $2c$.

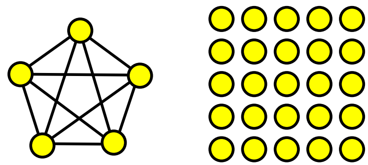
Ex 1. Maximum degree $\leqslant d \quad \Rightarrow \quad$ Average degree $\leqslant d$.

Ex 2. Planar graph has $\leqslant 3n - 6$ edges $\quad \Rightarrow \quad$ Average degree < 6.

Issue: A complete graph on k vertices plus k^2 isolated vertices.

– Average degree smaller than 1.

– Contains a **dense** subgraph.



Measuring sparsity

Measuring sparsity

Attempt 2. Every **subgraph** of G has a linear number of edges.

Measuring sparsity

Attempt 2. Every **subgraph** of G has a linear number of edges.

– We define **maximum average degree** of G as

$$\mathrm{mad}(G) \coloneqq \max_{H \subseteq G} \mathrm{avgdeg}(H).$$

Measuring sparsity

Attempt 2. Every **subgraph** of G has a linear number of edges.

– We define **maximum average degree** of G as

$$\mathrm{mad}(G) := \max_{H \subseteq G} \mathrm{avgdeg}(H).$$

– G is **sparse** if $\mathrm{mad}(G) \leqslant c$ for some constant c.

Measuring sparsity

Attempt 2. Every **subgraph** of G has a linear number of edges.

- We define **maximum average degree** of G as

$$\mathrm{mad}(G) \coloneqq \max_{H \subseteq G} \mathrm{avgdeg}(H).$$

- G is **sparse** if $\mathrm{mad}(G) \leqslant c$ for some constant c.

Ex 1. G has maximum degree $\leqslant d \quad \Rightarrow \quad \mathrm{mad}(G) \leqslant d$.

Measuring sparsity

Attempt 2. Every **subgraph** of G has a linear number of edges.

– We define **maximum average degree** of G as

$$\mathrm{mad}(G) \coloneqq \max_{H \subseteq G} \mathrm{avgdeg}(H).$$

– G is **sparse** if $\mathrm{mad}(G) \leqslant c$ for some constant c.

Ex 1. G has maximum degree $\leqslant d \quad \Rightarrow \quad \mathrm{mad}(G) \leqslant d$.

Ex 2. G is planar $\quad \Rightarrow \quad \mathrm{mad}(G) < 6$.

Measuring sparsity

Attempt 2. Every **subgraph** of G has a linear number of edges.

– We define **maximum average degree** of G as

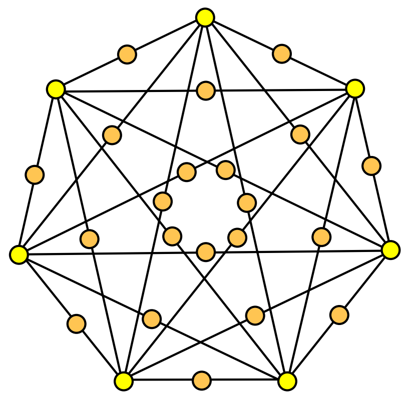
$$\mathrm{mad}(G) \coloneqq \max_{H \subseteq G} \mathrm{avgdeg}(H).$$

– G is **sparse** if $\mathrm{mad}(G) \leqslant c$ for some constant c.

Ex 1. G has maximum degree $\leqslant d \quad \Rightarrow \quad \mathrm{mad}(G) \leqslant d$.

Ex 2. G is planar $\quad \Rightarrow \quad \mathrm{mad}(G) < 6$.

Issue: A subdivided complete graph.



Measuring sparsity

Attempt 2. Every **subgraph** of G has a linear number of edges.

– We define **maximum average degree** of G as

$$\mathrm{mad}(G) := \max_{H \subseteq G} \mathrm{avgdeg}(H).$$

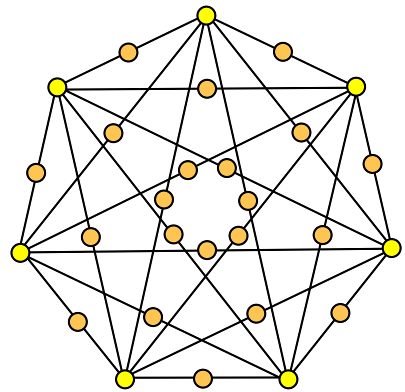
– G is **sparse** if $\mathrm{mad}(G) \leqslant c$ for some constant c.

Ex 1. G has maximum degree $\leqslant d \quad \Rightarrow \quad \mathrm{mad}(G) \leqslant d$.

Ex 2. G is planar $\quad \Rightarrow \quad \mathrm{mad}(G) < 6$.

Issue: A subdivided complete graph.

– Every subgraph has avgdeg $\leqslant 4$.



Measuring sparsity

Attempt 2. Every **subgraph** of G has a linear number of edges.

– We define **maximum average degree** of G as

$$\mathrm{mad}(G) := \max_{H \subseteq G} \mathrm{avgdeg}(H).$$

– G is **sparse** if $\mathrm{mad}(G) \leqslant c$ for some constant c.

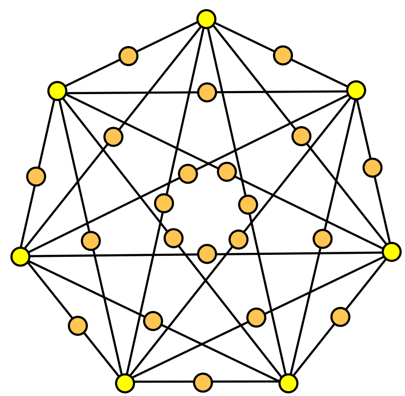
Ex 1. G has maximum degree $\leqslant d \quad \Rightarrow \quad \mathrm{mad}(G) \leqslant d$.

Ex 2. G is planar $\quad \Rightarrow \quad \mathrm{mad}(G) < 6$.

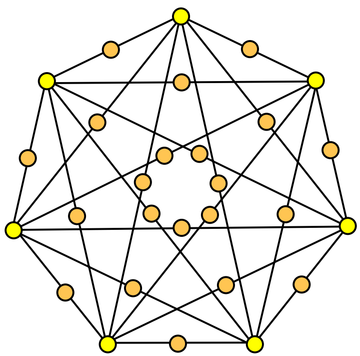
Issue: A subdivided complete graph.

– Every subgraph has avgdeg $\leqslant 4$.

– Is this graph really **sparse**?

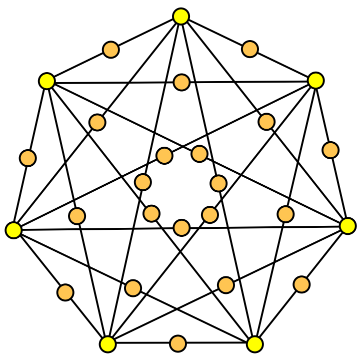


Measuring sparsity



Measuring sparsity

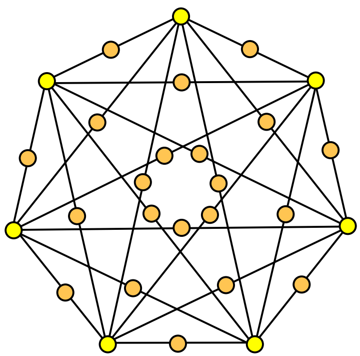
Option 1. We decide that a subdivided complete graph is **sparse**.



Measuring sparsity

Option 1. We decide that a subdivided complete graph is **sparse**.

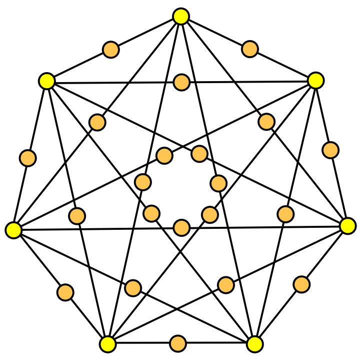
– We can construct a theory around the parameter $\mathrm{mad}(\cdot)$.

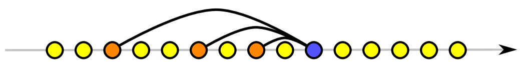


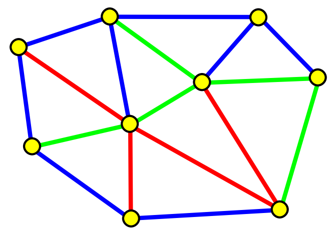
Measuring sparsity

Option 1. We decide that a subdivided complete graph is **sparse**.

- We can construct a theory around the parameter $\mathrm{mad}(\cdot)$.
- $\mathrm{mad}(\cdot)$ is essentially equivalent to **arboricity** and **degeneracy**.







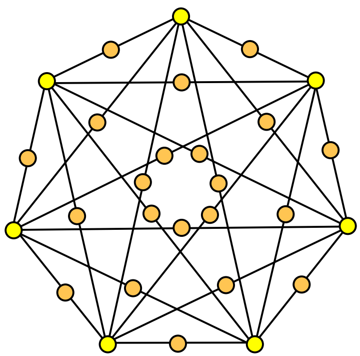
Measuring sparsity

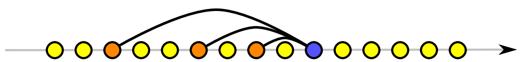
Option 1. We decide that a subdivided complete graph is **sparse**.

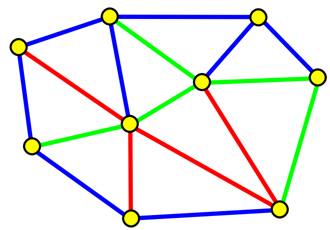
– We can construct a theory around the parameter $\mathrm{mad}(\cdot)$.

– $\mathrm{mad}(\cdot)$ is essentially equivalent to **arboricity** and **degeneracy**.

– These connections are useful, but not really very deep.





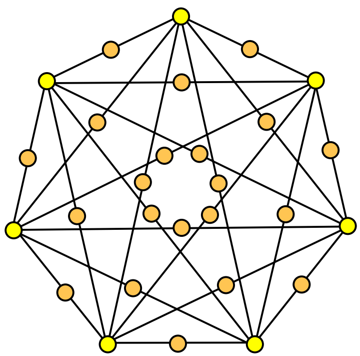


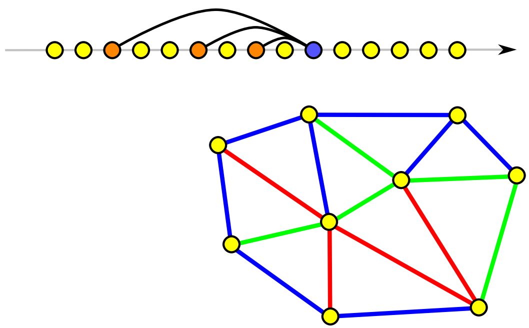
Measuring sparsity

Option 1. We decide that a subdivided complete graph is **sparse**.

- We can construct a theory around the parameter $\mathrm{mad}(\cdot)$.
- $\mathrm{mad}(\cdot)$ is essentially equivalent to **arboricity** and **degeneracy**.
- These connections are useful, but not really very deep.

Option 2. We decide that a subdivided complete graph is **dense**.





Measuring sparsity

Option 1. We decide that a subdivided complete graph is **sparse**.

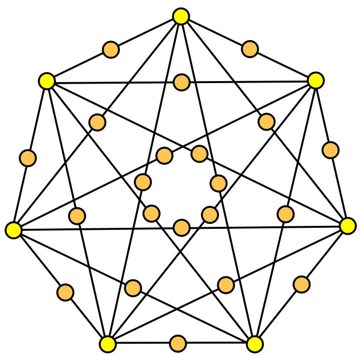
– We can construct a theory around the parameter $\mathrm{mad}(\cdot)$.

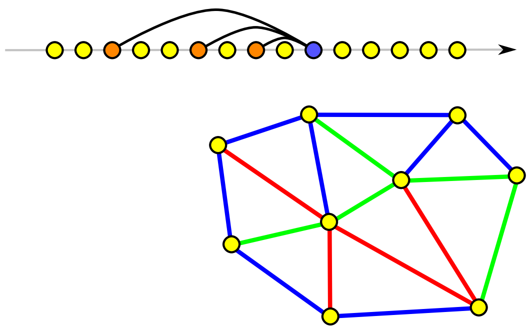
– $\mathrm{mad}(\cdot)$ is essentially equivalent to **arboricity** and **degeneracy**.

– These connections are useful, but not really very deep.

Option 2. We decide that a subdivided complete graph is **dense**.

– **Reason:** It contains a dense substructure visible at "depth" 1.





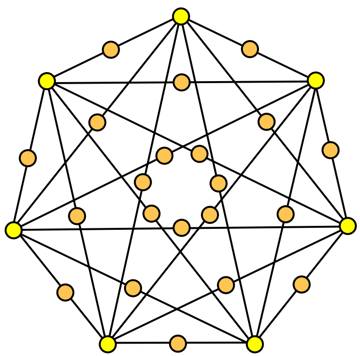
Measuring sparsity

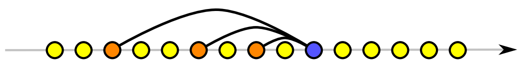
Option 1. We decide that a subdivided complete graph is **sparse**.

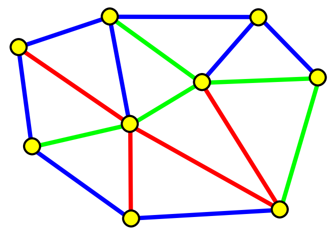
- We can construct a theory around the parameter $\mathrm{mad}(\cdot)$.
- $\mathrm{mad}(\cdot)$ is essentially equivalent to **arboricity** and **degeneracy**.
- These connections are useful, but not really very deep.

Option 2. We decide that a subdivided complete graph is **dense**.

- **Reason:** It contains a dense substructure visible at "depth" 1.
- **Need:** A notion of **embedding** that would capture this.

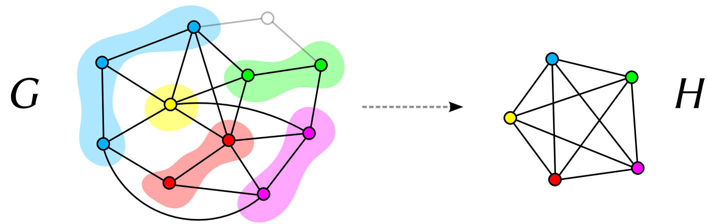






Minor order

Minor order

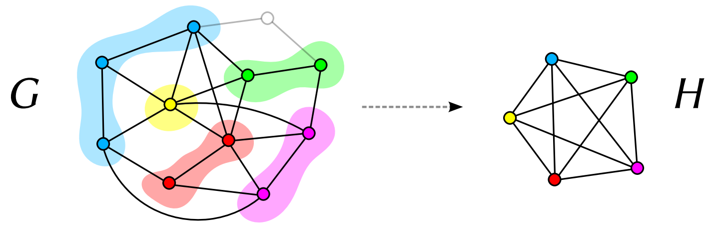


Definition

H is a **minor** of G $\Leftrightarrow$

H is obtained from a subgraph of G by contracting connected subgraphs

Minor order



Definition

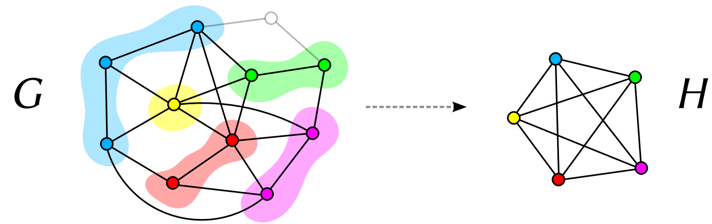
H is a **minor** of G $\Leftrightarrow$

H is obtained from a subgraph of G by contracting connected subgraphs

Theorem (Kuratowski; Wagner)

Planar graphs are exactly $\{K_5, K_{3,3}\}$-minor-free graphs.

Minor order



Definition

H is a **minor** of G $\Leftrightarrow$

H is obtained from a subgraph of G by contracting connected subgraphs

Theorem (Kuratowski; Wagner)

Planar graphs are exactly $\{K_5, K_{3,3}\}$-minor-free graphs.

Theorem (Robertson and Seymour)

For every $t \in \mathbb{N}$, every K_t-minor-free graph looks like this:

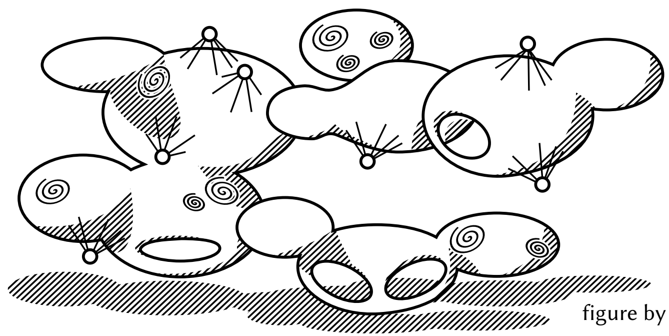


figure by Felix Reidl

Shallow minors

Attempt 3. Graphs excluding K_t as a minor, for some $t \in \mathbb{N}$.

Shallow minors

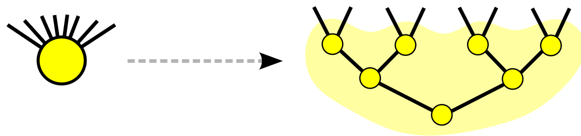
Attempt 3. Graphs excluding K_t as a minor, for some $t \in \mathbb{N}$.

Issue: Graphs with maxdeg 3 admit all complete graphs as minors.

Shallow minors

Attempt 3. Graphs excluding K_t as a minor, for some $t \in \mathbb{N}$.

Issue: Graphs with maxdeg 3 admit all complete graphs as minors.



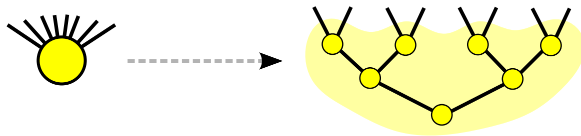
Shallow minors

Attempt 3. Graphs excluding K_t as a minor, for some $t \in \mathbb{N}$.

Issue: Graphs with maxdeg 3 admit all complete graphs as minors.

Ergo: **Excluding minors** leads to an interesting theory,

but this is **not** the theory we are after.



Shallow minors

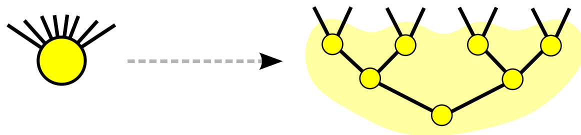
Attempt 3. Graphs excluding K_t as a minor, for some $t \in \mathbb{N}$.

Issue: Graphs with maxdeg 3 admit all complete graphs as minors.

Ergo: **Excluding minors** leads to an interesting theory,

but this is **not** the theory we are after.

Idea: Think about **local** minors.



Shallow minors

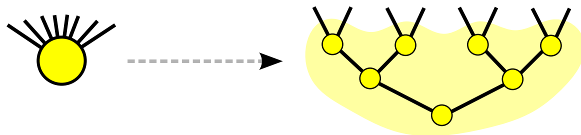
Attempt 3. Graphs excluding K_t as a minor, for some $t \in \mathbb{N}$.

Issue: Graphs with maxdeg 3 admit all complete graphs as minors.

Ergo: **Excluding minors** leads to an interesting theory,

but this is **not** the theory we are after.

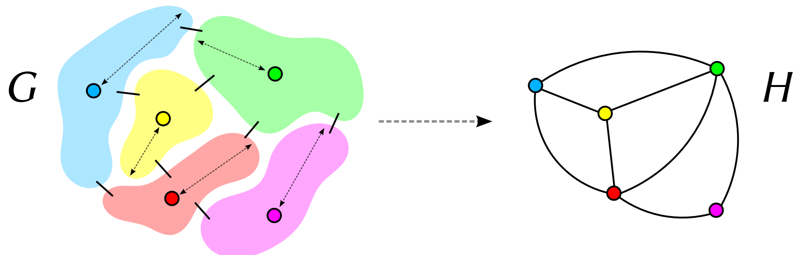
Idea: Think about **local** minors.



Definition

H is a **depth-d minor** of G $\Leftrightarrow$

H is obtained from a subgraph of G by contracting subgraphs of **radius** $\leqslant d$



Shallow minors

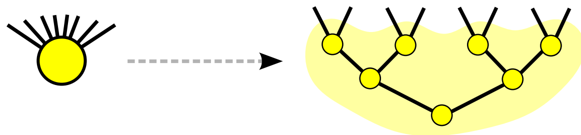
Attempt 3. Graphs excluding K_t as a minor, for some $t \in \mathbb{N}$.

Issue: Graphs with maxdeg 3 admit all complete graphs as minors.

Ergo: **Excluding minors** leads to an interesting theory,

but this is **not** the theory we are after.

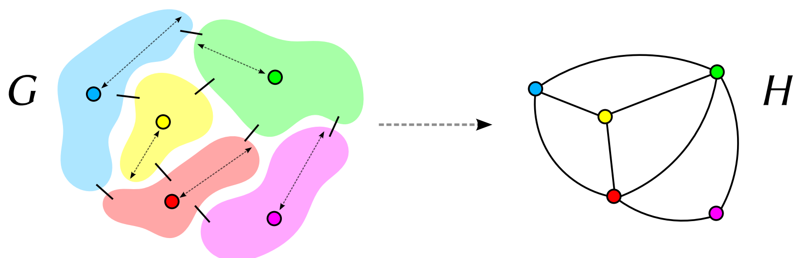
Idea: Think about **local** minors.



Definition

H is a **depth-d minor** of G $\Leftrightarrow$

H is obtained from a subgraph of G by contracting subgraphs of **radius** $\leqslant d$



depth-0 minors =

Shallow minors

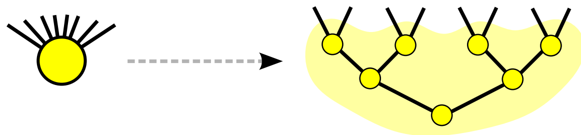
Attempt 3. Graphs excluding K_t as a minor, for some $t \in \mathbb{N}$.

Issue: Graphs with maxdeg 3 admit all complete graphs as minors.

Ergo: **Excluding minors** leads to an interesting theory,

but this is **not** the theory we are after.

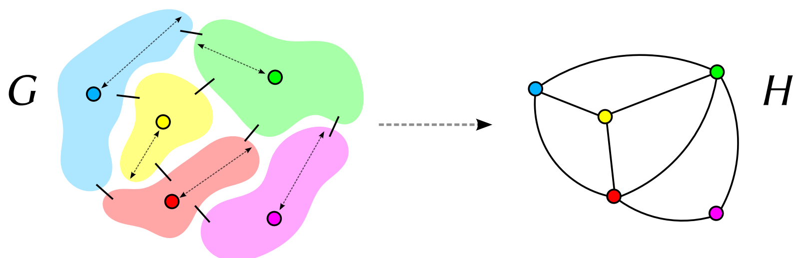
Idea: Think about **local** minors.



Definition

H is a **depth-d minor** of G $\Leftrightarrow$

H is obtained from a subgraph of G by contracting subgraphs of **radius** $\leqslant d$



depth-0 minors = subgraphs

Notions of sparsity

Notions of sparsity

Intuition: **Sparsity** $\Leftrightarrow$ Exclusion of **dense** structures at every fixed depth

Notions of sparsity

Intuition: **Sparsity** $\Leftrightarrow$ Exclusion of **dense** structures at every fixed depth

Definition

$$\begin{aligned} \nabla_d(G) &:= \mathsf{sup}\,\{\,\mathrm{avgdeg}(H) \,:\; H \text{ is a depth-}d \text{ minor of } G\,\} \\ \omega_d(G) &:= \mathsf{sup}\,\{\,t \,:\; K_t \text{ is a depth-}d \text{ minor of } G\,\}. \end{aligned}$$

Notions of sparsity

Intuition: Sparsity $\Leftrightarrow$ Exclusion of **dense** structures at every fixed depth

Definition

$$\begin{aligned}\nabla_d(G) &:= \mathsf{sup}\,\{\,\mathrm{avgdeg}(H) : H \text{ is a depth-}d\text{ minor of } G\,\}\\ \omega_d(G) &:= \mathsf{sup}\,\{\,t : K_t \text{ is a depth-}d\text{ minor of } G\,\}.\end{aligned}$$

Note: depth-0 minors = subgraphs $\quad\rightsquigarrow\quad \nabla_0(G) = \mathbf{?}$

Notions of sparsity

Intuition: **Sparsity** $\Leftrightarrow$ Exclusion of **dense** structures at every fixed depth

Definition

$$\begin{aligned}\nabla_d(G) &:= \sup \{ \operatorname{avgdeg}(H) : H \text{ is a depth-}d \text{ minor of } G \} \\ \omega_d(G) &:= \sup \{ t : K_t \text{ is a depth-}d \text{ minor of } G \}.\end{aligned}$$

Note: depth-0 minors = subgraphs $\rightsquigarrow$ $\nabla_0(G) = \operatorname{mad}(G)$.

Notions of sparsity

Intuition: **Sparsity** $\Leftrightarrow$ Exclusion of **dense** structures at every fixed depth

Definition

$$\begin{aligned}\nabla_d(G) &:= \sup \{ \mathrm{avgdeg}(H) : H \text{ is a depth-}d \text{ minor of } G \} \\ \omega_d(G) &:= \sup \{ t : K_t \text{ is a depth-}d \text{ minor of } G \}.\end{aligned}$$

Note: depth-0 minors = subgraphs $\rightsquigarrow$ $\nabla_0(G) = \mathrm{mad}(G)$.

For a **class** of graphs $\mathcal{C}$, we write:

$$\nabla_d(\mathcal{C}) := \sup_{G \in \mathcal{C}} \nabla_d(G) \qquad \text{and} \qquad \omega_d(\mathcal{C}) := \sup_{G \in \mathcal{C}} \omega_d(G).$$

Notions of sparsity

Intuition: **Sparsity** $\Leftrightarrow$ Exclusion of **dense** structures at every fixed depth

Definition

$$\begin{aligned}\nabla_d(G) &:= \mathsf{sup}\,\{\,\mathrm{avgdeg}(H)\ :\ H \text{ is a depth-}d \text{ minor of } G\,\}\\ \omega_d(G) &:= \mathsf{sup}\,\{\,t\ :\ K_t \text{ is a depth-}d \text{ minor of } G\,\}.\end{aligned}$$

Note: depth-0 minors = subgraphs $\rightsquigarrow$ $\nabla_0(G) = \mathrm{mad}(G)$.

For a **class** of graphs $\mathcal{C}$, we write:

$$\nabla_d(\mathcal{C}) := \sup_{G\in\mathcal{C}} \nabla_d(G) \qquad \text{and} \qquad \omega_d(\mathcal{C}) := \sup_{G\in\mathcal{C}} \omega_d(G).$$

Definition

$\mathcal{C}$ has **bounded expansion** if $\nabla_d(\mathcal{C})$ is finite for all $d \in \mathbb{N}$.

$\mathcal{C}$ is **nowhere dense** if $\omega_d(\mathcal{C})$ is finite for all $d \in \mathbb{N}$.

Notions of sparsity

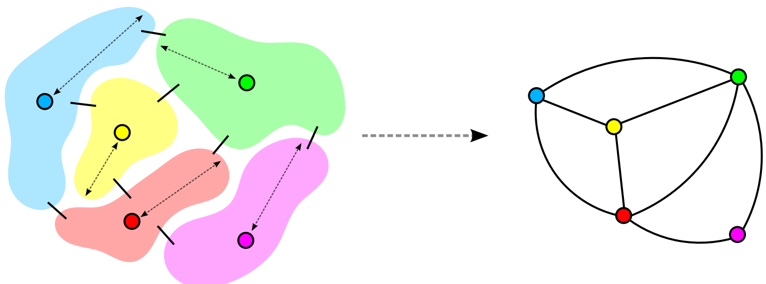
Equivalently:

Notions of sparsity

Equivalently:

$\mathcal{C}$ has **bounded expansion** if for every $d \in \mathbb{N}$ there is $c(d) \in \mathbb{N}$ s.t.

$\mathrm{avgdeg}(H) \leqslant c(d)$ whenever H is a depth-d minor of some $G \in \mathcal{C}$.

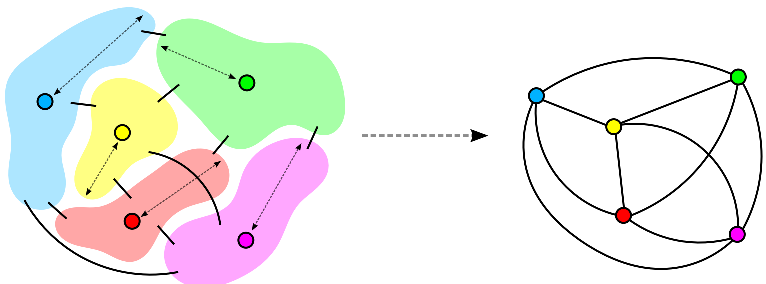


Notions of sparsity

Equivalently:

$\mathcal{C}$ has **bounded expansion** if for every $d \in \mathbb{N}$ there is $c(d) \in \mathbb{N}$ s.t.
$\mathrm{avgdeg}(H) \leqslant c(d)$ whenever H is a depth-d minor of some $G \in \mathcal{C}$.

$\mathcal{C}$ is **nowhere dense** if for every $d \in \mathbb{N}$ there is $t(d) \in \mathbb{N}$ s.t.
$K_{t(d)}$ is not a depth-d minor of any $G \in \mathcal{C}$.



Notions of sparsity

Equivalently:

$\mathcal{C}$ has **bounded expansion** if for every $d \in \mathbb{N}$ there is $c(d) \in \mathbb{N}$ s.t.
avgdeg$(H) \leqslant c(d)$ whenever H is a depth-d minor of some $G \in \mathcal{C}$.

$\mathcal{C}$ is **nowhere dense** if for every $d \in \mathbb{N}$ there is $t(d) \in \mathbb{N}$ s.t.
$K_{t(d)}$ is not a depth-d minor of any $G \in \mathcal{C}$.

Key idea: Sparsity is a property of a **class** of graphs.

Notions of sparsity

Equivalently:

$\mathcal{C}$ has **bounded expansion** if for every $d \in \mathbb{N}$ there is $c(d) \in \mathbb{N}$ s.t.

avgdeg$(H) \leqslant c(d)$ whenever H is a depth-d minor of some $G \in \mathcal{C}$.

$\mathcal{C}$ is **nowhere dense** if for every $d \in \mathbb{N}$ there is $t(d) \in \mathbb{N}$ s.t.

$K_{t(d)}$ is not a depth-d minor of any $G \in \mathcal{C}$.

Key idea: Sparsity is a property of a **class** of graphs.

– It is a **limit property** of graphs from the class.

Notions of sparsity

Equivalently:

$\mathcal{C}$ has **bounded expansion** if for every $d \in \mathbb{N}$ there is $c(d) \in \mathbb{N}$ s.t.

$\mathrm{avgdeg}(H) \leqslant c(d)$ whenever H is a depth-d minor of some $G \in \mathcal{C}$.

$\mathcal{C}$ is **nowhere dense** if for every $d \in \mathbb{N}$ there is $t(d) \in \mathbb{N}$ s.t.

$K_{t(d)}$ is not a depth-d minor of any $G \in \mathcal{C}$.

Key idea: Sparsity is a property of a **class** of graphs.

– It is a **limit property** of graphs from the class.

– Can be formalized using standard limit constructions (P, Toruńczyk).

Notions of sparsity

Equivalently:

$\mathcal{C}$ has **bounded expansion** if for every $d \in \mathbb{N}$ there is $c(d) \in \mathbb{N}$ s.t.

avgdeg$(H) \leqslant c(d)$ whenever H is a depth-d minor of some $G \in \mathcal{C}$.

$\mathcal{C}$ is **nowhere dense** if for every $d \in \mathbb{N}$ there is $t(d) \in \mathbb{N}$ s.t.

$K_{t(d)}$ is not a depth-d minor of any $G \in \mathcal{C}$.

Key idea: Sparsity is a property of a **class** of graphs.

- It is a **limit property** of graphs from the class.
- Can be formalized using standard limit constructions (P, Toruńczyk).
- **Classes** of graphs as basic objects of interest.

Examples and relations

1. Every class of **bounded degree** has **bounded expansion**.

Examples and relations

1. Every class of **bounded degree** has **bounded expansion**.

Sol: Depth-d minors have max degree $\leqslant \Delta^{d+1}$.

Examples and relations

1. Every class of **bounded degree** has **bounded expansion**.

Sol: Depth-d minors have max degree $\leqslant \Delta^{d+1}$.

2. Every class that **excludes some minor** has **bounded expansion**.

Examples and relations

1. Every class of **bounded degree** has **bounded expansion**.

   Sol: Depth-d minors have max degree $\leqslant \Delta^{d+1}$.

2. Every class that **excludes some minor** has **bounded expansion**.

   Sol: K_t-minor-free graphs have avgdeg $\mathcal{O}(t\sqrt{\log t})$ and are minor-closed.

Examples and relations

1. Every class of **bounded degree** has **bounded expansion**.

   Sol: Depth-d minors have max degree $\leqslant \Delta^{d+1}$.

2. Every class that **excludes some minor** has **bounded expansion**.

   Sol: K_t-minor-free graphs have avgdeg $\mathcal{O}(t\sqrt{\log t})$ and are minor-closed.

3. Every class of **bounded expansion** is **nowhere dense**.

Examples and relations

1. Every class of **bounded degree** has **bounded expansion**.

   Sol: Depth-d minors have max degree $\leqslant \Delta^{d+1}$.

2. Every class that **excludes some minor** has **bounded expansion**.

   Sol: K_t-minor-free graphs have avgdeg $\mathcal{O}(t\sqrt{\log t})$ and are minor-closed.

3. Every class of **bounded expansion** is **nowhere dense**.

   Sol: Cliques have unbounded average degree.

Examples and relations

1. Every class of **bounded degree** has **bounded expansion**.

   Sol: Depth-d minors have max degree $\leqslant \Delta^{d+1}$.

2. Every class that **excludes some minor** has **bounded expansion**.

   Sol: K_t-minor-free graphs have avgdeg $\mathcal{O}(t\sqrt{\log t})$ and are minor-closed.

3. Every class of **bounded expansion** is **nowhere dense**.

   Sol: Cliques have unbounded average degree.

4. Consider the class $\mathcal{C} = \{G \colon \Delta(G) \leqslant \mathrm{girth}(G)\}$.

Examples and relations

1. Every class of **bounded degree** has **bounded expansion**.

Sol: Depth-d minors have max degree $\leqslant \Delta^{d+1}$.

2. Every class that **excludes some minor** has **bounded expansion**.

Sol: K_t-minor-free graphs have avgdeg $\mathcal{O}(t\sqrt{\log t})$ and are minor-closed.

3. Every class of **bounded expansion** is **nowhere dense**.

Sol: Cliques have unbounded average degree.

4. Consider the class $\mathcal{C} = \{G \colon \Delta(G) \leqslant \mathrm{girth}(G)\}$.

4a: Show that $\mathcal{C}$ is **nowhere dense**.

Examples and relations

1. Every class of **bounded degree** has **bounded expansion**.

Sol: Depth-d minors have max degree $\leqslant \Delta^{d+1}$.

2. Every class that **excludes some minor** has **bounded expansion**.

Sol: K_t-minor-free graphs have avgdeg $\mathcal{O}(t\sqrt{\log t})$ and are minor-closed.

3. Every class of **bounded expansion** is **nowhere dense**.

Sol: Cliques have unbounded average degree.

4. Consider the class $\mathcal{C} = \{G \colon \Delta(G) \leqslant \mathrm{girth}(G)\}$.

4a: Show that $\mathcal{C}$ is **nowhere dense**.

Sol: $\omega_d(G) \geqslant 3 \Rightarrow d \geqslant \mathrm{girth}(G)/9 \Rightarrow d \geqslant \Delta(G)/9 \Rightarrow \omega_d(G) \leqslant (9d)^d$.

Examples and relations

1. Every class of **bounded degree** has **bounded expansion**.

Sol: Depth-d minors have max degree $\leqslant \Delta^{d+1}$.

2. Every class that **excludes some minor** has **bounded expansion**.

Sol: K_t-minor-free graphs have avgdeg $\mathcal{O}(t\sqrt{\log t})$ and are minor-closed.

3. Every class of **bounded expansion** is **nowhere dense**.

Sol: Cliques have unbounded average degree.

4. Consider the class $\mathcal{C} = \{G \colon \Delta(G) \leqslant \mathrm{girth}(G)\}$.

4a: Show that $\mathcal{C}$ is **nowhere dense**.

Sol: $\omega_d(G) \geqslant 3 \Rightarrow d \geqslant \mathrm{girth}(G)/9 \Rightarrow d \geqslant \Delta(G)/9 \Rightarrow \omega_d(G) \leqslant (9d)^d$.

4b: Show that $\mathcal{C}$ does **not** have **bounded expansion**.

Examples and relations

1. Every class of **bounded degree** has **bounded expansion**.

Sol: Depth-d minors have max degree $\leqslant \Delta^{d+1}$.

2. Every class that **excludes some minor** has **bounded expansion**.

Sol: K_t-minor-free graphs have avgdeg $\mathcal{O}(t\sqrt{\log t})$ and are minor-closed.

3. Every class of **bounded expansion** is **nowhere dense**.

Sol: Cliques have unbounded average degree.

4. Consider the class $\mathcal{C} = \{G\colon \Delta(G) \leqslant \mathrm{girth}(G)\}$.

4a: Show that $\mathcal{C}$ is **nowhere dense**.

Sol: $\omega_d(G) \geqslant 3 \Rightarrow d \geqslant \mathrm{girth}(G)/9 \Rightarrow d \geqslant \Delta(G)/9 \Rightarrow \omega_d(G) \leqslant (9d)^d$.

4b: Show that $\mathcal{C}$ does **not** have **bounded expansion**.

Sol: Erdős random construction.

Examples and relations

1. Every class of **bounded degree** has **bounded expansion**.

Sol: Depth-d minors have max degree $\leqslant \Delta^{d+1}$.

2. Every class that **excludes some minor** has **bounded expansion**.

Sol: K_t-minor-free graphs have avgdeg $\mathcal{O}(t\sqrt{\log t})$ and are minor-closed.

3. Every class of **bounded expansion** is **nowhere dense**.

Sol: Cliques have unbounded average degree.

4. Consider the class $\mathcal{C} = \{G \colon \Delta(G) \leqslant \mathrm{girth}(G)\}$.

4a: Show that $\mathcal{C}$ is **nowhere dense**.

Sol: $\omega_d(G) \geqslant 3 \Rightarrow d \geqslant \mathrm{girth}(G)/9 \Rightarrow d \geqslant \Delta(G)/9 \Rightarrow \omega_d(G) \leqslant (9d)^d$.

4b: Show that $\mathcal{C}$ does **not** have **bounded expansion**.

Sol: Erdős random construction.

Fact. $\mathcal{C}$ is **nowhere dense** $\Rightarrow \forall d, \varepsilon > 0$, $\nabla_d(G) \leqslant \mathcal{O}(n^{\varepsilon})$ for all $G \in \mathcal{C}$.

The World of Sparsity

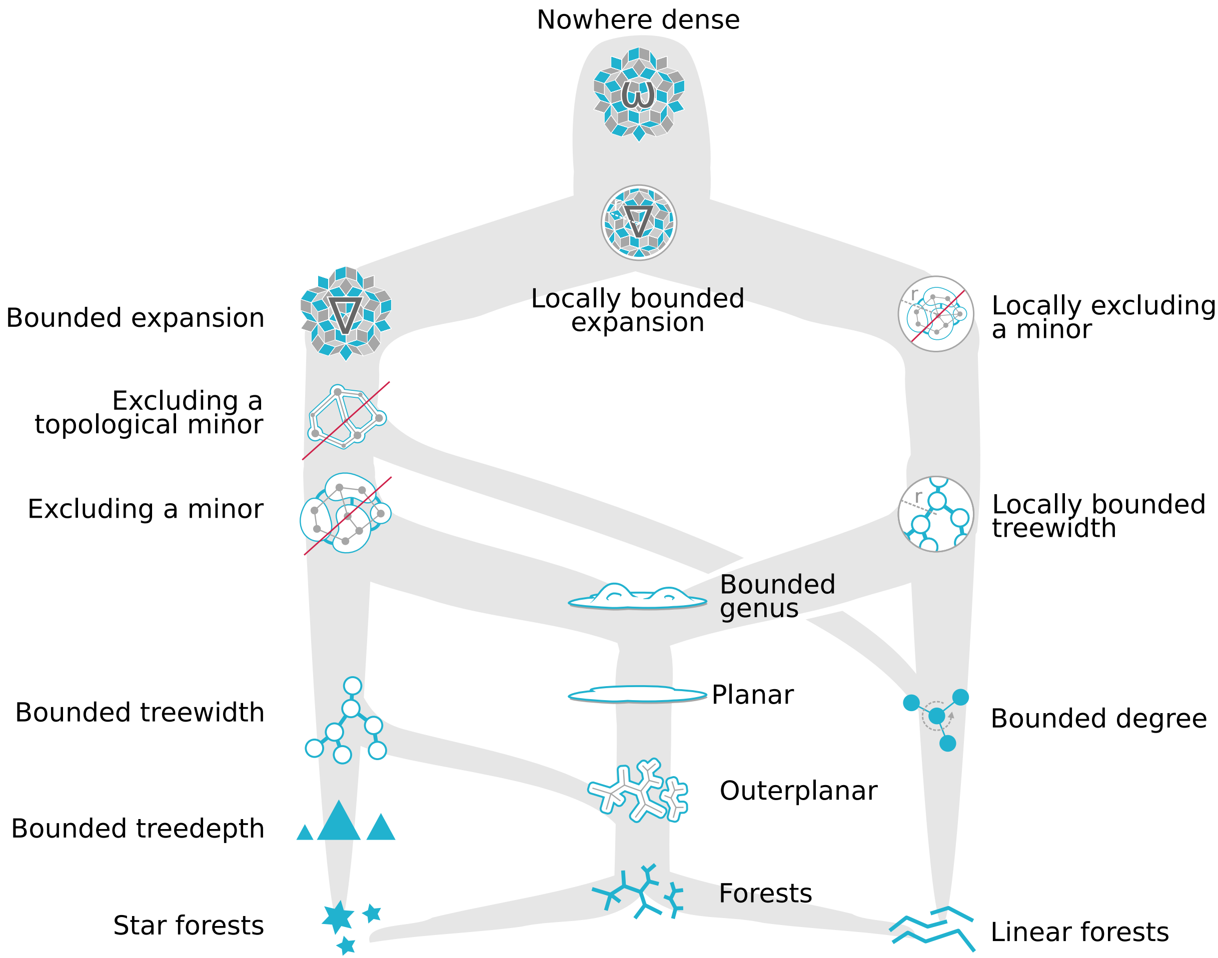
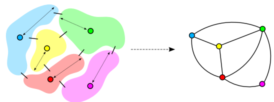


figure by Felix Reidl

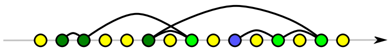
Equivalent characterizations

Sparsity of shallow minors

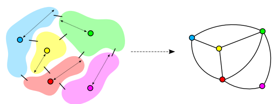


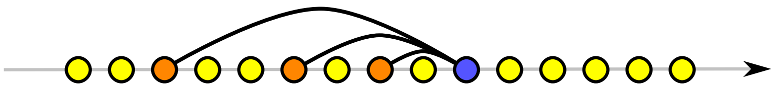
Equivalent characterizations

Generalized coloring numbers

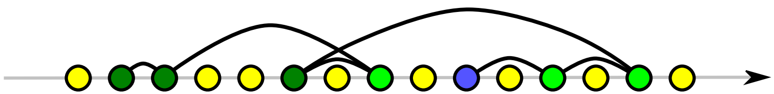


Sparsity of shallow minors





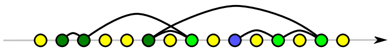
Degeneracy



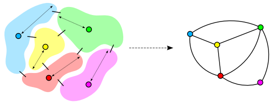
Weak coloring number

Equivalent characterizations

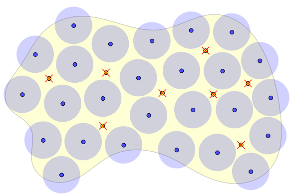
Generalized coloring numbers

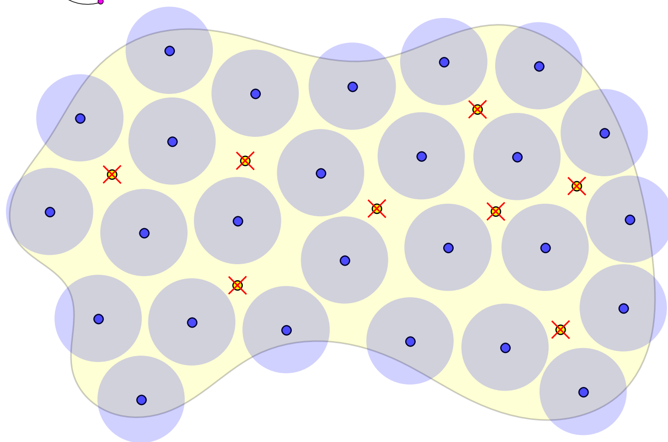


Sparsity of shallow minors



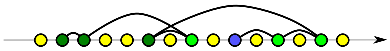
Uniform quasi-wideness

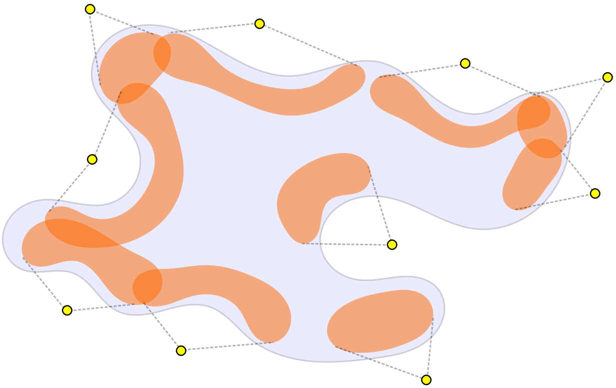




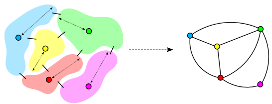
Equivalent characterizations

Generalized coloring numbers

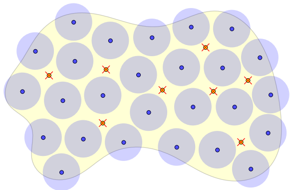




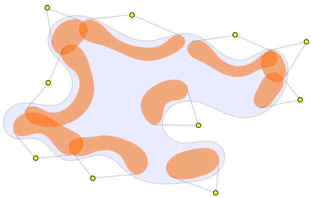
Sparsity of shallow minors



Uniform quasi-wideness



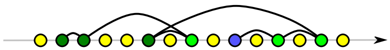
Neighborhood complexity



Equivalent characterizations

Part 2

Generalized coloring numbers



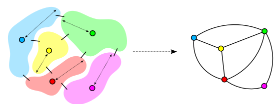
Stability

Sparsity of shallow top-minors

Fraternal augmentations

Neighborhood covers

Sparsity of shallow minors



Part 3

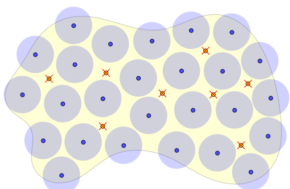
Low treedepth colorings

Uniform quasi-wideness

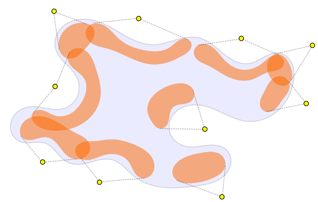
k-Helly property

Neighborhood complexity

Part 4



Splitter game



Working with Sparsity

Many characterizations of **bnd expansion** and **nowhere denseness**.

Working with Sparsity

Many characterizations of **bnd expansion** and **nowhere denseness**.

– Equivalence shows that we are working with fundamental notions.

Working with Sparsity

Many characterizations of **bnd expansion** and **nowhere denseness**.

- Equivalence shows that we are working with fundamental notions.
- Each characterization is a **tool**.

Working with Sparsity

Many characterizations of **bnd expansion** and **nowhere denseness**.

- Equivalence shows that we are working with fundamental notions.
- Each characterization is a **tool**.

Original idea: Study the **combinatorics** of sparse graphs.

Working with Sparsity

Many characterizations of **bnd expansion** and **nowhere denseness**.

- Equivalence shows that we are working with fundamental notions.
- Each characterization is a **tool**.

Original idea: Study the **combinatorics** of sparse graphs.

- **Goal:** Describe structural properties implied by sparsity.

Working with Sparsity

Many characterizations of **bnd expansion** and **nowhere denseness**.

- Equivalence shows that we are working with fundamental notions.
- Each characterization is a **tool**.

Original idea: Study the **combinatorics** of sparse graphs.

- **Goal:** Describe structural properties implied by sparsity.

These properties can be used to design efficient **algorithms**.

Working with Sparsity

Many characterizations of **bnd expansion** and **nowhere denseness**.

- Equivalence shows that we are working with fundamental notions.
- Each characterization is a **tool**.

Original idea: Study the **combinatorics** of sparse graphs.

- **Goal:** Describe structural properties implied by sparsity.

These properties can be used to design efficient **algorithms**.

- **Areas:** Parameterized, approximation, and distributed algorithms.

Working with Sparsity

Many characterizations of **bnd expansion** and **nowhere denseness**.

– Equivalence shows that we are working with fundamental notions.

– Each characterization is a **tool**.

Original idea: Study the **combinatorics** of sparse graphs.

– **Goal:** Describe structural properties implied by sparsity.

These properties can be used to design efficient **algorithms**.

– **Areas:** Parameterized, approximation, and distributed algorithms.

– Applicable to problems of **local** nature.

Working with Sparsity

Many characterizations of **bnd expansion** and **nowhere denseness**.

- Equivalence shows that we are working with fundamental notions.
- Each characterization is a **tool**.

Original idea: Study the **combinatorics** of sparse graphs.

- **Goal:** Describe structural properties implied by sparsity.

These properties can be used to design efficient **algorithms**.

- **Areas:** Parameterized, approximation, and distributed algorithms.
- Applicable to problems of **local** nature.

Sparsity delimits tractability of **First Order logic** on graphs.

Working with Sparsity

Many characterizations of **bnd expansion** and **nowhere denseness**.

- Equivalence shows that we are working with fundamental notions.
- Each characterization is a **tool**.

Original idea: Study the **combinatorics** of sparse graphs.

- **Goal:** Describe structural properties implied by sparsity.

These properties can be used to design efficient **algorithms**.

- **Areas:** Parameterized, approximation, and distributed algorithms.
- Applicable to problems of **local** nature.

Sparsity delimits tractability of **First Order logic** on graphs.

Theorem (Grohe, Kreutzer, Siebertz)

Suppose $\mathcal{C}$ is subgraph-closed.

Then $\mathcal{C}$ is **nowhere dense** $\Leftrightarrow$ FO Model Checking is FPT on $\mathcal{C}$.

Working with Sparsity

Many characterizations of **bnd expansion** and **nowhere denseness**.

- Equivalence shows that we are working with fundamental notions.
- Each characterization is a **tool**.

Original idea: Study the **combinatorics** of sparse graphs.

- **Goal:** Describe structural properties implied by sparsity.

These properties can be used to design efficient **algorithms**.

- **Areas:** Parameterized, approximation, and distributed algorithms.
- Applicable to problems of **local** nature.

Sparsity delimits tractability of **First Order logic** on graphs.

Theorem (Grohe, Kreutzer, Siebertz)

Suppose $\mathcal{C}$ is subgraph-closed.

Then $\mathcal{C}$ is **nowhere dense** $\Leftrightarrow$ FO Model Checking is FPT on $\mathcal{C}$.

Part 2:

Generalized coloring numbers

Degeneracy

Definition

G is **d-degenerate** $\Leftrightarrow$ Every subgraph of G has a vertex of degree $\leqslant d$.

$\mathrm{dgn}(G) :=$ least d for which G is d-degenerate.

Degeneracy

Definition

G is **d-degenerate** $\Leftrightarrow$ Every subgraph of G has a vertex of degree $\leqslant d$.

$\mathrm{dgn}(G) :=$ least d for which G is d-degenerate.

1. Prove that $\mathrm{mad}(G)/2 \leqslant \mathrm{dgn}(G) \leqslant \mathrm{mad}(G)$.

Degeneracy

Definition

G is **d-degenerate** $\Leftrightarrow$ Every subgraph of G has a vertex of degree $\leqslant d$.

$\mathrm{dgn}(G) :=$ least d for which G is d-degenerate.

1. Prove that $\mathrm{mad}(G)/2 \leqslant \mathrm{dgn}(G) \leqslant \mathrm{mad}(G)$.

right: $\quad \mathrm{mindeg} \leqslant \mathrm{avgdeg}$

Degeneracy

Definition

G is ***d*-degenerate** $\Leftrightarrow$ Every subgraph of G has a vertex of degree $\leqslant d$.

$\mathrm{dgn}(G) :=$ least d for which G is d-degenerate.

1. Prove that $\mathrm{mad}(G)/2 \leqslant \mathrm{dgn}(G) \leqslant \mathrm{mad}(G)$.

right: $\quad \mathrm{mindeg} \leqslant \mathrm{avgdeg}$

left: $\quad |E(H)| \leqslant \mathrm{dgn}(G) \cdot |V(H)|$ by removing vertices one by one.

Degeneracy

Definition

G is ***d*-degenerate** $\Leftrightarrow$ Every subgraph of G has a vertex of degree $\leqslant d$.

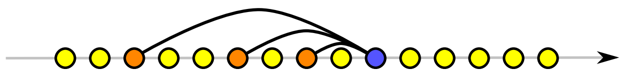
$\mathrm{dgn}(G) :=$ least d for which G is d-degenerate.

1. Prove that $\mathrm{mad}(G)/2 \leqslant \mathrm{dgn}(G) \leqslant \mathrm{mad}(G)$.

right: $\mathrm{mindeg} \leqslant \mathrm{avgdeg}$

left: $|E(H)| \leqslant \mathrm{dgn}(G) \cdot |V(H)|$ by removing vertices one by one.

2. Prove that G is d-degenerate $\Leftrightarrow$



G has a **vertex ordering** where each vertex has $\leqslant d$ neighbors to the left.

Degeneracy

Definition

G is **d-degenerate** $\Leftrightarrow$ Every subgraph of G has a vertex of degree $\leqslant d$.

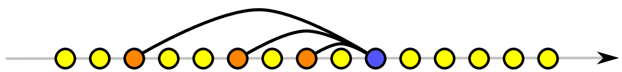
$\mathrm{dgn}(G) :=$ least d for which G is d-degenerate.

1. Prove that $\mathrm{mad}(G)/2 \leqslant \mathrm{dgn}(G) \leqslant \mathrm{mad}(G)$.

right: $\quad \mathrm{mindeg} \leqslant \mathrm{avgdeg}$

left: $\quad |E(H)| \leqslant \mathrm{dgn}(G) \cdot |V(H)|$ by removing vertices one by one.

2. Prove that G is d-degenerate $\Leftrightarrow$



G has a **vertex ordering** where each vertex has $\leqslant d$ neighbors to the left.

($\Rightarrow$): $\quad$ extract vertices one by one

Degeneracy

Definition

G is ***d*-degenerate** $\Leftrightarrow$ Every subgraph of G has a vertex of degree $\leqslant d$.

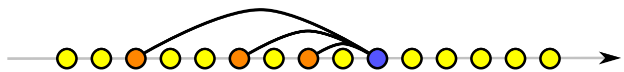
$\mathrm{dgn}(G)$:= least d for which G is d-degenerate.

1. Prove that $\mathrm{mad}(G)/2 \leqslant \mathrm{dgn}(G) \leqslant \mathrm{mad}(G)$.

right: $\mathrm{mindeg} \leqslant \mathrm{avgdeg}$

left: $|E(H)| \leqslant \mathrm{dgn}(G) \cdot |V(H)|$ by removing vertices one by one.

2. Prove that G is d-degenerate $\Leftrightarrow$



G has a **vertex ordering** where each vertex has $\leqslant d$ neighbors to the left.

($\Rightarrow$): extract vertices one by one

($\Leftarrow$): examine the rightmost vertex

Degeneracy

Definition

G is ***d*-degenerate** $\Leftrightarrow$ Every subgraph of G has a vertex of degree $\leqslant d$.

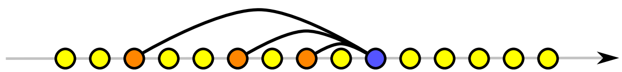
$\mathrm{dgn}(G) :=$ least d for which G is d-degenerate.

1. Prove that $\mathrm{mad}(G)/2 \leqslant \mathrm{dgn}(G) \leqslant \mathrm{mad}(G)$.

right: $\mathrm{mindeg} \leqslant \mathrm{avgdeg}$

left: $|E(H)| \leqslant \mathrm{dgn}(G) \cdot |V(H)|$ by removing vertices one by one.

2. Prove that G is d-degenerate $\Leftrightarrow$



G has a **vertex ordering** where each vertex has $\leqslant d$ neighbors to the left.

($\Rightarrow$): extract vertices one by one

($\Leftarrow$): examine the rightmost vertex

3. d-degenerate graphs are $(d+1)$-colorable

Degeneracy

Definition

G is ***d*-degenerate** $\Leftrightarrow$ Every subgraph of G has a vertex of degree $\leqslant d$.

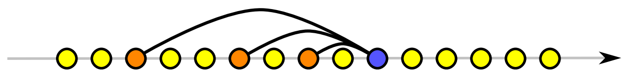
$\mathrm{dgn}(G) :=$ least d for which G is d-degenerate.

1. Prove that $\mathrm{mad}(G)/2 \leqslant \mathrm{dgn}(G) \leqslant \mathrm{mad}(G)$.

right: $\quad$ mindeg $\leqslant$ avgdeg

left: $\quad |E(H)| \leqslant \mathrm{dgn}(G) \cdot |V(H)|$ by removing vertices one by one.

2. Prove that G is d-degenerate $\Leftrightarrow$



G has a **vertex ordering** where each vertex has $\leqslant d$ neighbors to the left.

($\Rightarrow$): $\quad$ extract vertices one by one

($\Leftarrow$): $\quad$ examine the rightmost vertex

3. d-degenerate graphs are $(d+1)$-colorable

Sol: Greedy left-to-right coloring on the ordering.

Generalizing degeneracy

Generalizing degeneracy

Idea: A degeneracy ordering exposes a **global** structure in a graph.

Generalizing degeneracy

Idea: A degeneracy ordering exposes a **global** structure in a graph.

Degeneracy orderings $\leftrightsquigarrow$ $\mathrm{mad}(\cdot)$, which concerns depth-0 minors.

Generalizing degeneracy

Idea: A degeneracy ordering exposes a **global** structure in a graph.

Degeneracy orderings $\leftrightsquigarrow$ $\mathrm{mad}(\cdot)$, which concerns depth-0 minors.

Idea: Introduce a generalization of degeneracy orderings to larger depth.

Generalizing degeneracy

Idea: A degeneracy ordering exposes a **global** structure in a graph.

Degeneracy orderings $\leftrightsquigarrow$ mad$(\cdot)$, which concerns depth-0 minors.

Idea: Introduce a generalization of degeneracy orderings to larger depth.

These are called **generalized coloring numbers**.

Generalizing degeneracy

Idea: A degeneracy ordering exposes a **global** structure in a graph.

Degeneracy orderings $\leftrightsquigarrow$ $\mathrm{mad}(\cdot)$, which concerns depth-0 minors.

Idea: Introduce a generalization of degeneracy orderings to larger depth.

These are called **generalized coloring numbers**.

There are **three** natural ways to make the generalization.

Generalizing degeneracy

Idea: A degeneracy ordering exposes a **global** structure in a graph.

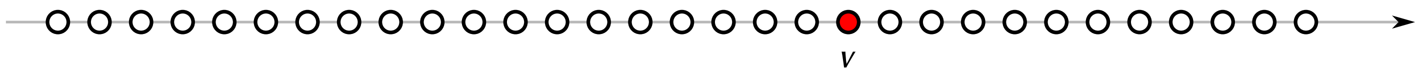
Degeneracy orderings $\leftrightsquigarrow$ $\mathrm{mad}(\cdot)$, which concerns depth-0 minors.

Idea: Introduce a generalization of degeneracy orderings to larger depth.

These are called **generalized coloring numbers**.

There are **three** natural ways to make the generalization.

Suppose $d \in \mathbb{N}$, G is a graph, and σ is a vertex ordering.



Generalizing degeneracy

Idea: A degeneracy ordering exposes a **global** structure in a graph.

Degeneracy orderings $\leftrightsquigarrow$ $\mathrm{mad}(\cdot)$, which concerns depth-0 minors.

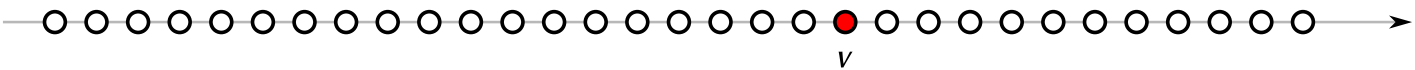
Idea: Introduce a generalization of degeneracy orderings to larger depth.

These are called **generalized coloring numbers**.

There are **three** natural ways to make the generalization.

Suppose $d \in \mathbb{N}$, G is a graph, and σ is a vertex ordering.

Consider any vertex v.



Generalizing degeneracy

Idea: A degeneracy ordering exposes a **global** structure in a graph.

Degeneracy orderings $\leftrightsquigarrow$ $\mathrm{mad}(\cdot)$, which concerns depth-0 minors.

Idea: Introduce a generalization of degeneracy orderings to larger depth.

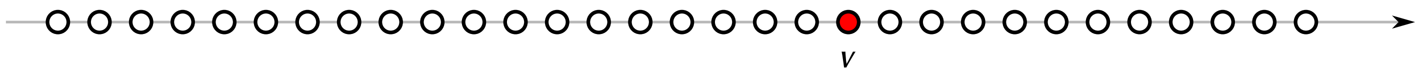
These are called **generalized coloring numbers**.

There are **three** natural ways to make the generalization.

Suppose $d \in \mathbb{N}$, G is a graph, and σ is a vertex ordering.

Consider any vertex v.

Want: Define "σ-smaller neighbors" of v at "depth" d.

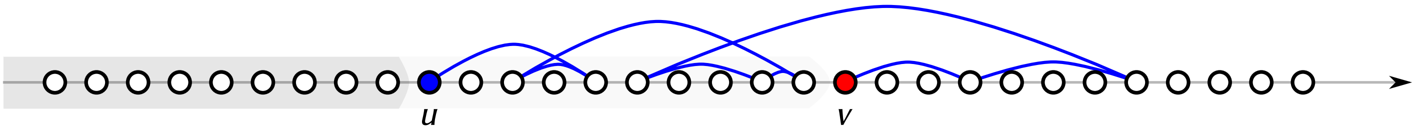


Bounded depth reachability

Bounded depth reachability

D1: $u \leqslant_\sigma v$ is **weakly d-reachable** from v $\Leftrightarrow$

There is a v-to-u path P of length $\leqslant d$ that is entirely $\geqslant_\sigma u$.

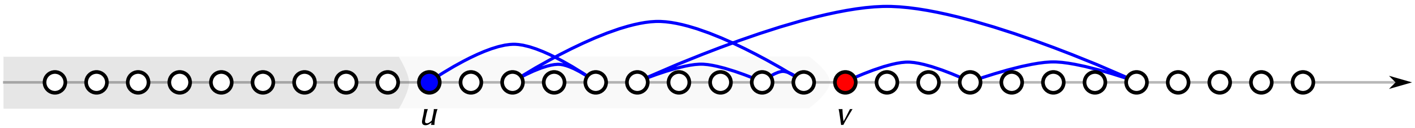


Bounded depth reachability

D1: $u \leqslant_\sigma v$ is **weakly d-reachable** from v $\Leftrightarrow$

There is a v-to-u path P of length $\leqslant d$ that is entirely $\geqslant_\sigma u$.

$\mathrm{WReach}_d[G, \sigma, v] \coloneqq \{\, u \leqslant_\sigma v : u \text{ is weakly } d\text{-reachable from } v \,\}.$

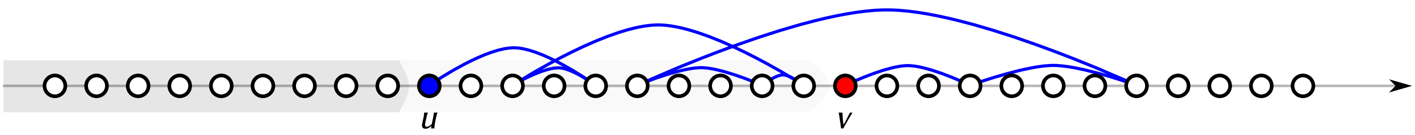


Bounded depth reachability

D1: $u \leqslant_\sigma v$ is **weakly d-reachable** from v $\Leftrightarrow$

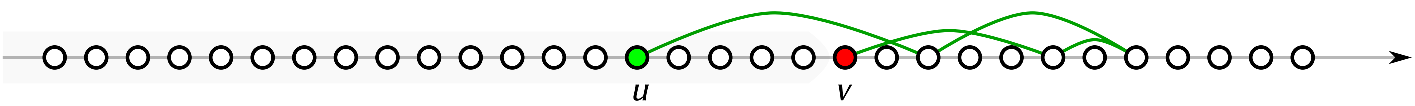
There is a v-to-u path P of length $\leqslant d$ that is entirely $\geqslant_\sigma u$.

$\mathrm{WReach}_d[G, \sigma, v] := \{\, u \leqslant_\sigma v : u$ is weakly d-reachable from $v\,\}$.



D2: $u \leqslant_\sigma v$ is **strongly d-reachable** from v $\Leftrightarrow$

There is a v-to-u path P of length $\leqslant d$ that is $\geqslant_\sigma v$, apart from u

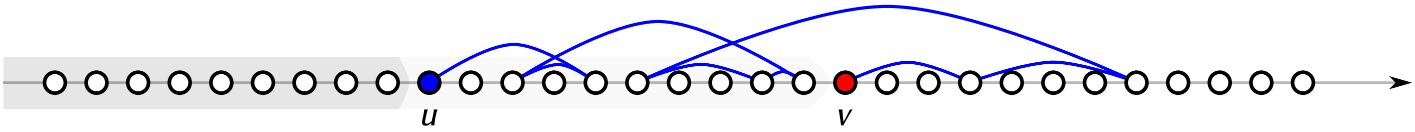


Bounded depth reachability

D1: $u \leqslant_\sigma v$ is **weakly d-reachable** from $v \quad \Leftrightarrow$

There is a v-to-u path P of length $\leqslant d$ that is entirely $\geqslant_\sigma u$.

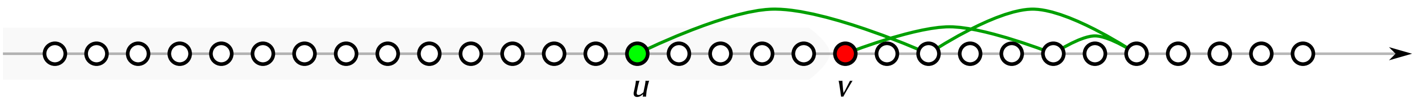
$$\mathrm{WReach}_d[G,\sigma,v] := \{\, u \leqslant_\sigma v : \ u \text{ is weakly } d\text{-reachable from } v \,\}.$$



D2: $u \leqslant_\sigma v$ is **strongly d-reachable** from $v \quad \Leftrightarrow$

There is a v-to-u path P of length $\leqslant d$ that is $\geqslant_\sigma v$, apart from u

$$\mathrm{SReach}_d[G,\sigma,v] := \{\, u \leqslant_\sigma v : \ u \text{ is strongly } d\text{-reachable from } v \,\}.$$

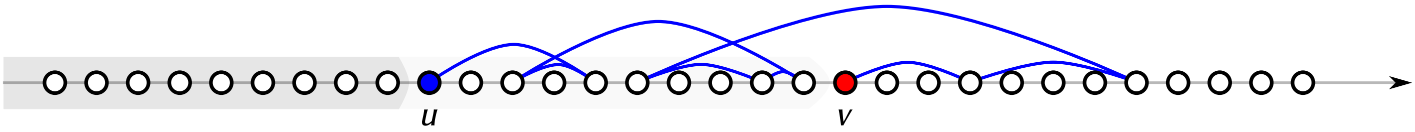


Bounded depth reachability

D1: $u \leqslant_\sigma v$ is **weakly d-reachable** from $v \quad \Leftrightarrow$

There is a v-to-u path P of length $\leqslant d$ that is entirely $\geqslant_\sigma u$.

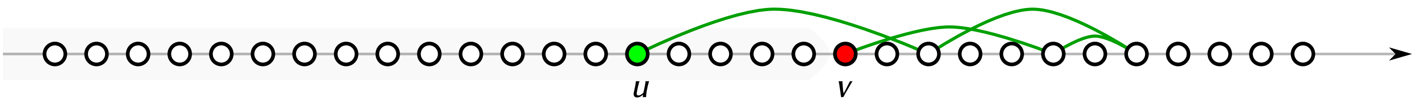
$\mathrm{WReach}_d[G, \sigma, v] := \{\, u \leqslant_\sigma v : \; u \text{ is weakly } d\text{-reachable from } v \,\}$.



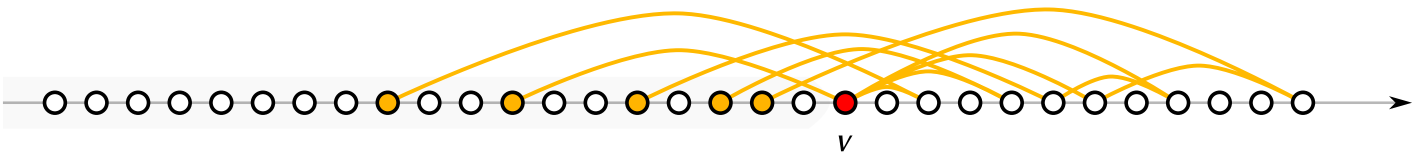
D2: $u \leqslant_\sigma v$ is **strongly d-reachable** from $v \quad \Leftrightarrow$

There is a v-to-u path P of length $\leqslant d$ that is $\geqslant_\sigma v$, apart from u

$\mathrm{SReach}_d[G, \sigma, v] := \{\, u \leqslant_\sigma v : \; u \text{ is strongly } d\text{-reachable from } v \,\}$.



D3: $\mathrm{adm}_d(G, \sigma, v)$:=max # of disjoint v-to-($<_\sigma v$) paths of length $\leqslant d$



Generalized coloring numbers

Generalized coloring numbers

Definition

For a graph G and vertex ordering σ, we define:

$$\mathrm{wcol}_d(G,\sigma) := \max_v \; |\mathrm{WReach}_d[G,\sigma,v]|,$$

$$\mathrm{scol}_d(G,\sigma) := \max_v \; |\mathrm{SReach}_d[G,\sigma,v]|,$$

$$\mathrm{adm}_d(G,\sigma) := \max_v \; \mathrm{adm}_d(G,\sigma,v),$$

Generalized coloring numbers

Definition

For a graph G and vertex ordering σ, we define:

$$\mathrm{wcol}_d(G,\sigma) \coloneqq \max_v \ |\mathrm{WReach}_d[G,\sigma,v]|, \qquad \mathrm{wcol}_d(G) \coloneqq \min_\sigma \ \mathrm{wcol}_d(G,\sigma),$$

$$\mathrm{scol}_d(G,\sigma) \coloneqq \max_v \ |\mathrm{SReach}_d[G,\sigma,v]|, \qquad \mathrm{scol}_d(G) \coloneqq \min_\sigma \ \mathrm{scol}_d(G,\sigma),$$

$$\mathrm{adm}_d(G,\sigma) \coloneqq \max_v \ \mathrm{adm}_d(G,\sigma,v), \qquad \mathrm{adm}_d(G) \coloneqq \min_\sigma \ \mathrm{adm}_d(G,\sigma).$$

Generalized coloring numbers

Definition

For a graph G and vertex ordering σ, we define:

$$\mathrm{wcol}_d(G,\sigma) \coloneqq \max_v \ |\mathrm{WReach}_d[G,\sigma,v]|, \qquad \mathrm{wcol}_d(G) \coloneqq \min_\sigma \ \mathrm{wcol}_d(G,\sigma),$$

$$\mathrm{scol}_d(G,\sigma) \coloneqq \max_v \ |\mathrm{SReach}_d[G,\sigma,v]|, \qquad \mathrm{scol}_d(G) \coloneqq \min_\sigma \ \mathrm{scol}_d(G,\sigma),$$

$$\mathrm{adm}_d(G,\sigma) \coloneqq \max_v \ \mathrm{adm}_d(G,\sigma,v), \qquad \mathrm{adm}_d(G) \coloneqq \min_\sigma \ \mathrm{adm}_d(G,\sigma).$$

1. $\mathrm{dgn}(G) = \mathrm{adm}_0(G) = \mathrm{scol}_0(G) - 1 = \mathrm{wcol}_0(G) - 1.$

Generalized coloring numbers

Definition

For a graph G and vertex ordering σ, we define:

$$\mathrm{wcol}_d(G,\sigma) := \max_v \, |\mathrm{WReach}_d[G,\sigma,v]|, \qquad \mathrm{wcol}_d(G) := \min_\sigma \, \mathrm{wcol}_d(G,\sigma),$$
$$\mathrm{scol}_d(G,\sigma) := \max_v \, |\mathrm{SReach}_d[G,\sigma,v]|, \qquad \mathrm{scol}_d(G) := \min_\sigma \, \mathrm{scol}_d(G,\sigma),$$
$$\mathrm{adm}_d(G,\sigma) := \max_v \, \mathrm{adm}_d(G,\sigma,v), \qquad \mathrm{adm}_d(G) := \min_\sigma \, \mathrm{adm}_d(G,\sigma).$$

1. $\mathrm{dgn}(G) = \mathrm{adm}_0(G) = \mathrm{scol}_0(G) - 1 = \mathrm{wcol}_0(G) - 1.$

2. $\mathrm{adm}_d(G) \leqslant \mathrm{scol}_d(G) \leqslant \mathrm{wcol}_d(G).$

Generalized coloring numbers

Definition

For a graph G and vertex ordering σ, we define:

$$\mathrm{wcol}_d(G,\sigma) := \max_v \ |\mathrm{WReach}_d[G,\sigma,v]|, \qquad \mathrm{wcol}_d(G) := \min_\sigma \ \mathrm{wcol}_d(G,\sigma),$$

$$\mathrm{scol}_d(G,\sigma) := \max_v \ |\mathrm{SReach}_d[G,\sigma,v]|, \qquad \mathrm{scol}_d(G) := \min_\sigma \ \mathrm{scol}_d(G,\sigma),$$

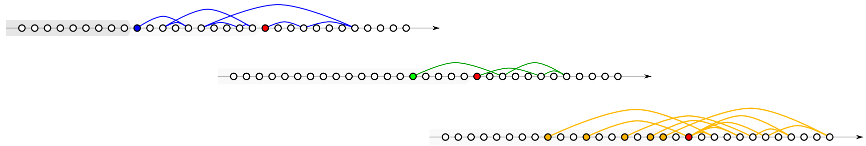
$$\mathrm{adm}_d(G,\sigma) := \max_v \ \mathrm{adm}_d(G,\sigma,v), \qquad \mathrm{adm}_d(G) := \min_\sigma \ \mathrm{adm}_d(G,\sigma).$$

1. $\mathrm{dgn}(G) = \mathrm{adm}_0(G) = \mathrm{scol}_0(G) - 1 = \mathrm{wcol}_0(G) - 1.$

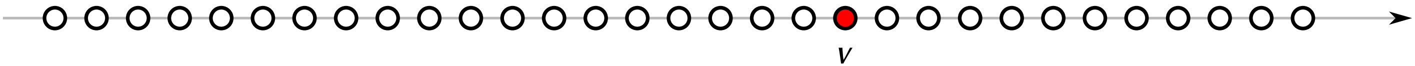
2. $\mathrm{adm}_d(G) \leqslant \mathrm{scol}_d(G) \leqslant \mathrm{wcol}_d(G).$

Now: These parameters are functionally equivalent.

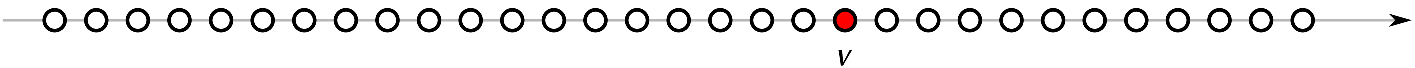
Equivalence of generalized coloring numbers



1. $\mathrm{scol}_d(G) \leqslant 1 + \mathrm{adm}_d(G)^d$.



2. $\mathrm{wcol}_d(G) \leqslant 1 + \mathrm{scol}_d(G) + \mathrm{scol}_d(G)^2 + \ldots + \mathrm{scol}_d(G)^d$.



Equivalence with grads

Lemma

For a graph G and $d \in \mathbb{N}$, we have:

$$\mathrm{adm}_d(G) \leqslant 6d(\nabla_d(G) + 1)^3, \qquad \nabla_d(G) \leqslant \mathrm{wcol}_{4d+1}(G).$$

Equivalence with grads

Lemma

For a graph G and $d \in \mathbb{N}$, we have:

$$\mathrm{adm}_d(G) \leqslant 6d(\nabla_d(G)+1)^3, \qquad \nabla_d(G) \leqslant \mathrm{wcol}_{4d+1}(G).$$

Proof of $\nabla_d(G) \leqslant \mathrm{wcol}_{4d+1}(G)$:

Equivalence with grads

Lemma

For a graph G and $d \in \mathbb{N}$, we have:

$$\mathrm{adm}_d(G) \leqslant 6d(\nabla_d(G) + 1)^3, \qquad \nabla_d(G) \leqslant \mathrm{wcol}_{4d+1}(G).$$

Proof of $\nabla_d(G) \leqslant \mathrm{wcol}_{4d+1}(G)$:

- Let $H \preceq_d G$ and $\{J_u \colon u \in V(H)\}$ be a model.
- Let $\phi(u) \coloneqq \min_\sigma V(J_u)$.
- Let $w \in V(H)$ be such that $\phi(w)$ is σ-maximal.
- **Obs:** For each $u \in N_H(w)$, we have $\phi(u) \in \mathrm{WReach}_{4d+1}[G, \sigma, \phi(w)]$.
- **Cor:** w has degree $\leqslant \mathrm{wcol}_{4d+1}(G)$ in H.
- **Cor:** Every $H \preceq_d G$ has a vertex of degree $\leqslant \mathrm{wcol}_{4d+1}(G)$. $\square$

Equivalence with grads

Equivalence with grads

Sketch of proof of $\mathrm{adm}_d(G) \leqslant 6d(\nabla_d(G)+1)^3$**:**

Equivalence with grads

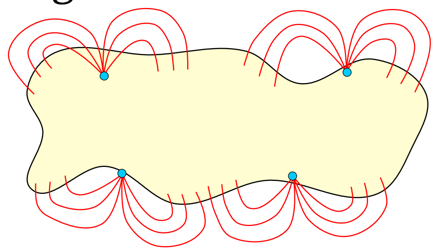
Sketch of proof of $\mathrm{adm}_d(G) \leqslant 6d(\nabla_d(G)+1)^3$**:**

– There is a greedy algorithm, similarly as for degeneracy.

Equivalence with grads

Sketch of proof of $\mathrm{adm}_d(G) \leqslant 6d(\nabla_d(G)+1)^3$**:**

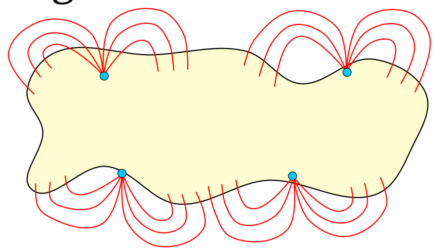
- There is a greedy algorithm, similarly as for degeneracy.
- If the algorithm gets stuck, it uncovers the following structure:



Equivalence with grads

Sketch of proof of $\mathrm{adm}_d(G) \leqslant 6d(\nabla_d(G)+1)^3$**:**

- There is a greedy algorithm, similarly as for degeneracy.
- If the algorithm gets stuck, it uncovers the following structure:



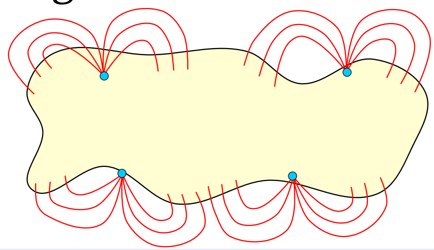
- We can now find a dense depth-d minor. $\square$

Equivalence with grads

Sketch of proof of $\mathrm{adm}_d(G) \leqslant 6d(\nabla_d(G)+1)^3$**:**

– There is a greedy algorithm, similarly as for degeneracy.

– If the algorithm gets stuck, it uncovers the following structure:



– We can now find a dense depth-d minor. $\square$

Theorem

For a class of graphs $\mathcal{C}$, the following are equivalent:

– $\mathcal{C}$ has bounded expansion;

– $\nabla_d(\mathcal{C})$ is finite for all $d \in \mathbb{N}$;

– $\mathrm{wcol}_d(\mathcal{C})$ is finite for all $d \in \mathbb{N}$;

– $\mathrm{scol}_d(\mathcal{C})$ is finite for all $d \in \mathbb{N}$;

– $\mathrm{adm}_d(\mathcal{C})$ is finite for all $d \in \mathbb{N}$.

Distance-d domination

Definition

Let G be a graph and $d \in \mathbb{N}$.

$D \subseteq V(G)$ is a **dist-d dominating set** if $\bigcup_{u \in D} \mathrm{Ball}_d(u) = V(G)$.

Distance-d domination

Definition

Let G be a graph and $d \in \mathbb{N}$.

$D \subseteq V(G)$ is a **dist-d dominating set** if $\bigcup_{u \in D} \mathrm{Ball}_d(u) = V(G)$.

$\mathrm{dom}_d(G) :=$ min size of a dist-d dominating set in G

Distance-d domination

Definition

Let G be a graph and $d \in \mathbb{N}$.

$D \subseteq V(G)$ is a **dist-d dominating set** if $\bigcup_{u \in D} \mathrm{Ball}_d(u) = V(G)$.

$\mathrm{dom}_d(G) :=$ min size of a dist-d dominating set in G

Let σ be a vertex ordering of G. Consider the algorithm:

Distance-d domination

Definition

Let G be a graph and $d \in \mathbb{N}$.

$D \subseteq V(G)$ is a **dist-d dominating set** if $\bigcup_{u \in D} \mathrm{Ball}_d(u) = V(G)$.

$\mathrm{dom}_d(G) \coloneqq$ min size of a dist-d dominating set in G

Let σ be a vertex ordering of G. Consider the algorithm:

– Every vertex v **points** to $\min_\sigma \mathrm{WReach}_d[G, \sigma, v]$.

Distance-d domination

Definition

Let G be a graph and $d \in \mathbb{N}$.

$D \subseteq V(G)$ is a **dist-d dominating set** if $\bigcup_{u \in D} \mathrm{Ball}_d(u) = V(G)$.

$\mathrm{dom}_d(G) :=$ min size of a dist-d dominating set in G

Let σ be a vertex ordering of G. Consider the algorithm:

- Every vertex v **points** to $\min_\sigma \mathrm{WReach}_d[G, \sigma, v]$.
- Let $D :=$ set of vertices that are **pointed** to.

Distance-d domination

Definition

Let G be a graph and $d \in \mathbb{N}$.

$D \subseteq V(G)$ is a **dist-d dominating set** if $\bigcup_{u \in D} \mathrm{Ball}_d(u) = V(G)$.

$\mathrm{dom}_d(G) :=$ min size of a dist-d dominating set in G

Let σ be a vertex ordering of G. Consider the algorithm:

– Every vertex v **points** to $\min_\sigma \mathrm{WReach}_d[G, \sigma, v]$.

– Let $D :=$ set of vertices that are **pointed** to.

1. Prove that $|D| \leqslant \mathrm{wcol}_{2d}(G, \sigma) \cdot \mathrm{dom}_d(G)$.

Hitting and packing duality

Hitting and packing duality

Cor: Constant-factor apx algorithm for $\mathrm{dom}_d(G)$ in bnd expansion classes.

Hitting and packing duality

Cor: Constant-factor apx algorithm for $\mathrm{dom}_d(G)$ in bnd expansion classes.

– Compute σ with $\mathrm{wcol}_{2d}(G, \sigma)$ bounded by a constant.

Hitting and packing duality

Cor: Constant-factor apx algorithm for $\mathrm{dom}_d(G)$ in bnd expansion classes.

- Compute σ with $\mathrm{wcol}_{2d}(G, \sigma)$ bounded by a constant.
- Each v picks $\min_\sigma \mathrm{WReach}_d[G, \sigma, v]$.

Hitting and packing duality

Cor: Constant-factor apx algorithm for $\mathrm{dom}_d(G)$ in bnd expansion classes.

- Compute σ with $\mathrm{wcol}_{2d}(G, \sigma)$ bounded by a constant.
- Each v picks $\min_\sigma \mathrm{WReach}_d[G, \sigma, v]$.
- Output the picked vertices.

Hitting and packing duality

Cor: Constant-factor apx algorithm for $\mathrm{dom}_d(G)$ in bnd expansion classes.

- Compute σ with $\mathrm{wcol}_{2d}(G, \sigma)$ bounded by a constant.
- Each v picks $\min_\sigma \mathrm{WReach}_d[G, \sigma, v]$.
- Output the picked vertices.

Obs: $\mathrm{dom}_d(G)$ is the **hitting number** for radius-d balls.

Hitting and packing duality

Cor: Constant-factor apx algorithm for $\mathrm{dom}_d(G)$ in bnd expansion classes.

- Compute σ with $\mathrm{wcol}_{2d}(G, \sigma)$ bounded by a constant.
- Each v picks $\min_\sigma \mathrm{WReach}_d[G, \sigma, v]$.
- Output the picked vertices.

Obs: $\mathrm{dom}_d(G)$ is the **hitting number** for radius-d balls.

Def: Let $\mathrm{sca}_d(G)$ be the **packing number** for radius-d balls.

Hitting and packing duality

Cor: Constant-factor apx algorithm for $\mathrm{dom}_d(G)$ in bnd expansion classes.

- Compute σ with $\mathrm{wcol}_{2d}(G,\sigma)$ bounded by a constant.
- Each v picks $\min_\sigma \mathrm{WReach}_d[G,\sigma,v]$.
- Output the picked vertices.

Obs: $\mathrm{dom}_d(G)$ is the **hitting number** for radius-d balls.

Def: Let $\mathrm{sca}_d(G)$ be the **packing number** for radius-d balls.

Obviously $\mathrm{sca}_d(G) \leqslant \mathrm{dom}_d(G)$, but the gap is unbounded in general.

Hitting and packing duality

Cor: Constant-factor apx algorithm for $\mathrm{dom}_d(G)$ in bnd expansion classes.

- Compute σ with $\mathrm{wcol}_{2d}(G,\sigma)$ bounded by a constant.
- Each v picks $\min_\sigma \mathrm{WReach}_d[G,\sigma,v]$.
- Output the picked vertices.

Obs: $\mathrm{dom}_d(G)$ is the **hitting number** for radius-d balls.

Def: Let $\mathrm{sca}_d(G)$ be the **packing number** for radius-d balls.

Obviously $\mathrm{sca}_d(G) \leqslant \mathrm{dom}_d(G)$, but the gap is unbounded in general.

Theorem (Dvořák)

For every G and $d \in \mathbb{N}$, we have $\mathrm{dom}_d(G) \leqslant \mathrm{wcol}_{2d}(G)^2 \cdot \mathrm{sca}_d(G)$.

Hitting and packing duality

Cor: Constant-factor apx algorithm for $\mathrm{dom}_d(G)$ in bnd expansion classes.

– Compute σ with $\mathrm{wcol}_{2d}(G,\sigma)$ bounded by a constant.

– Each v picks $\min_\sigma \mathrm{WReach}_d[G,\sigma,v]$.

– Output the picked vertices.

Obs: $\mathrm{dom}_d(G)$ is the **hitting number** for radius-d balls.

Def: Let $\mathrm{sca}_d(G)$ be the **packing number** for radius-d balls.

Obviously $\mathrm{sca}_d(G) \leqslant \mathrm{dom}_d(G)$, but the gap is unbounded in general.

Theorem (Dvořák)

For every G and $d \in \mathbb{N}$, we have $\mathrm{dom}_d(G) \leqslant \mathrm{wcol}_{2d}(G)^2 \cdot \mathrm{sca}_d(G)$.

Cor: Constant-factor gap in bounded expansion classes.

Hitting and packing duality

Cor: Constant-factor apx algorithm for $\mathrm{dom}_d(G)$ in bnd expansion classes.

- Compute σ with $\mathrm{wcol}_{2d}(G,\sigma)$ bounded by a constant.
- Each v picks $\min_\sigma \mathrm{WReach}_d[G,\sigma,v]$.
- Output the picked vertices.

Obs: $\mathrm{dom}_d(G)$ is the **hitting number** for radius-d balls.

Def: Let $\mathrm{sca}_d(G)$ be the **packing number** for radius-d balls.

Obviously $\mathrm{sca}_d(G) \leqslant \mathrm{dom}_d(G)$, but the gap is unbounded in general.

Theorem (Dvořák)

For every G and $d \in \mathbb{N}$, we have $\mathrm{dom}_d(G) \leqslant \mathrm{wcol}_{2d}(G)^2 \cdot \mathrm{sca}_d(G)$.

Cor: Constant-factor gap in bounded expansion classes.

Proof: A greedy procedure on a vertex ordering witnessing $\mathrm{wcol}_{2d}(G)$.

Part 3:

Treedepth and low treedepth colorings

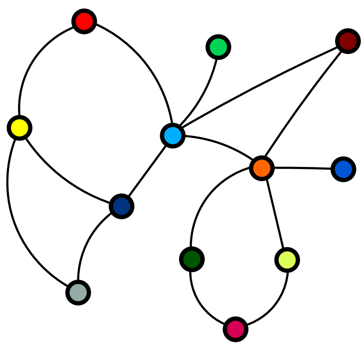
Treedepth

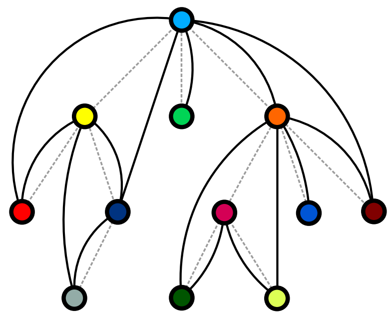
Treedepth

Definition

Elimination forest of G is a rooted forest F with $V(F) = V(G)$ s.t:

$$u, v \text{ adjacent in } G \quad \Rightarrow \quad u, v \text{ in ancestor/descendant relation in } F$$





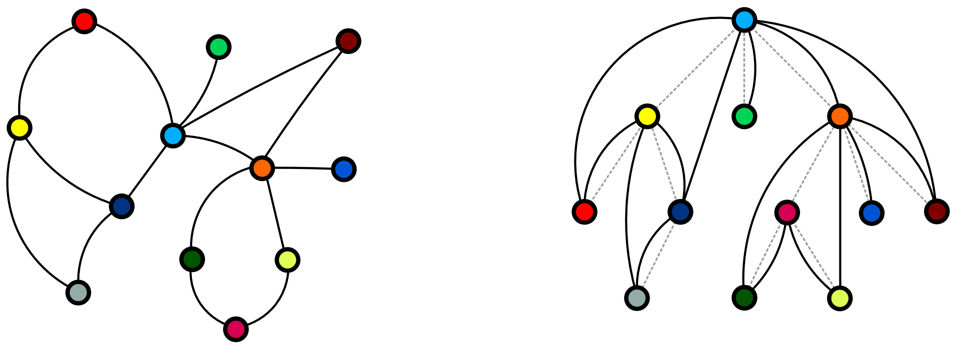
Treedepth

Definition

Elimination forest of G is a rooted forest F with $V(F) = V(G)$ s.t:

u, v adjacent in $G \quad \Rightarrow \quad u, v$ in ancestor/descendant relation in F

$\mathrm{td}(G) :=$ least possible depth of an elimination forest of G.



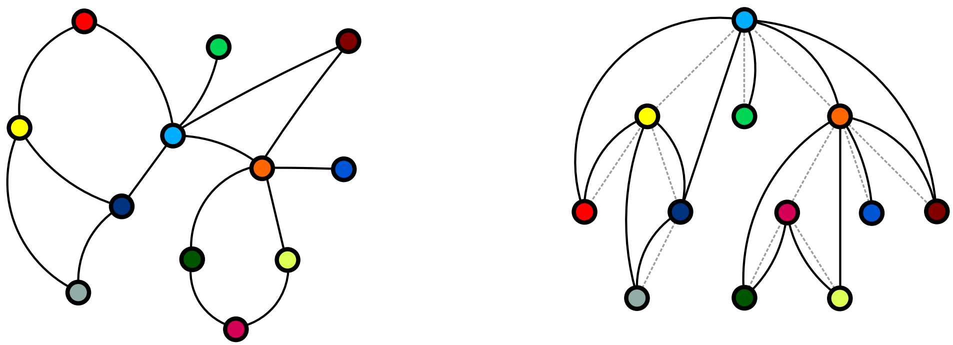
Treedepth

Definition

Elimination forest of G is a rooted forest F with $V(F) = V(G)$ s.t:

u, v adjacent in $G \quad \Rightarrow \quad u, v$ in ancestor/descendant relation in F

$\mathrm{td}(G) :=$ least possible depth of an elimination forest of G.



Basic remarks:

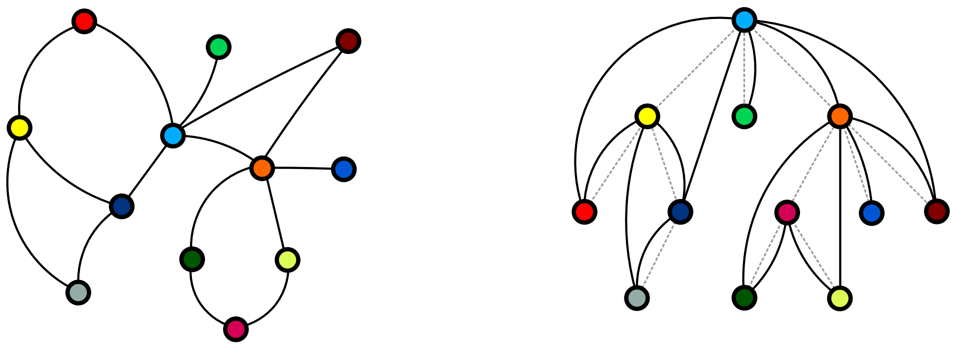
Treedepth

Definition

Elimination forest of G is a rooted forest F with $V(F) = V(G)$ s.t:

u, v adjacent in $G \quad \Rightarrow \quad u, v$ in ancestor/descendant relation in F

$\mathrm{td}(G) :=$ least possible depth of an elimination forest of G.



Basic remarks:

– If G is connected, then F must be a tree.

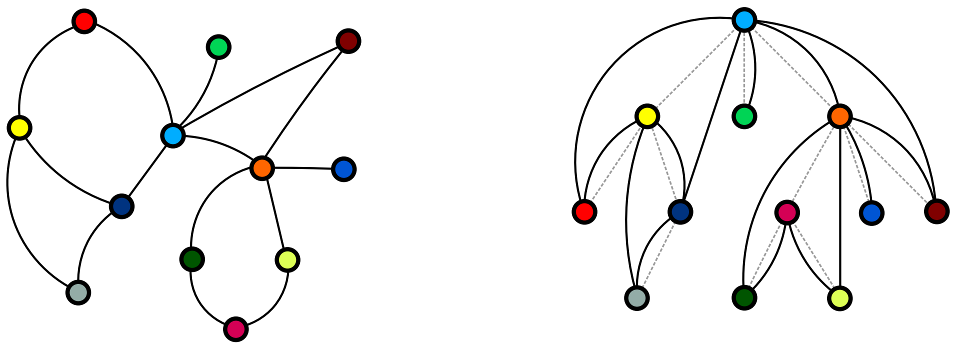
Treedepth

Definition

Elimination forest of G is a rooted forest F with $V(F) = V(G)$ s.t:

u, v adjacent in $G \quad \Rightarrow \quad u, v$ in ancestor/descendant relation in F

$\mathrm{td}(G) :=$ least possible depth of an elimination forest of G.



Basic remarks:

– If G is connected, then F must be a tree.

– $H \subseteq G \quad \Rightarrow \quad \mathrm{td}(H) \leqslant \mathrm{td}(G)$

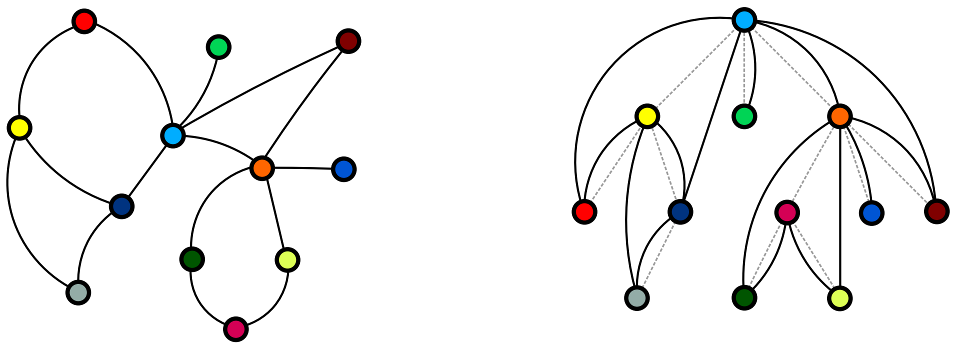
Treedepth

Definition

Elimination forest of G is a rooted forest F with $V(F) = V(G)$ s.t:

u, v adjacent in $G \quad \Rightarrow \quad u, v$ in ancestor/descendant relation in F

$\mathrm{td}(G) :=$ least possible depth of an elimination forest of G.



Basic remarks:

– If G is connected, then F must be a tree.

– $H \subseteq G \quad \Rightarrow \quad \mathrm{td}(H) \leqslant \mathrm{td}(G)$

– $\mathrm{tw}(G) \leqslant \mathrm{td}(G) \leqslant \mathrm{tw}(G) \cdot \log n$

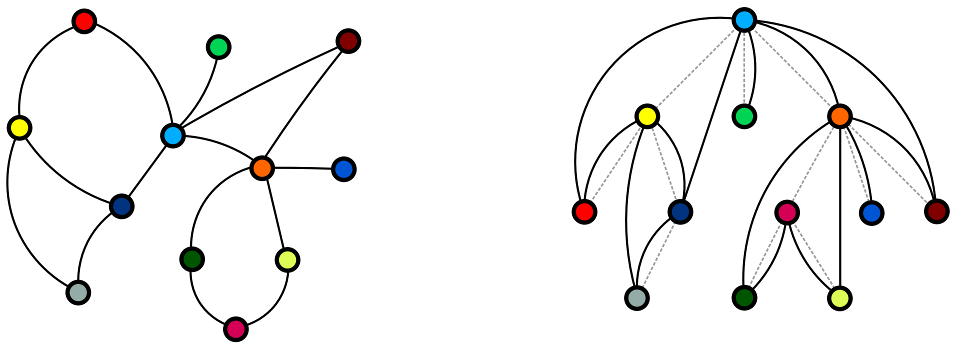
Treedepth

Definition

Elimination forest of G is a rooted forest F with $V(F) = V(G)$ s.t:

u, v adjacent in $G \quad \Rightarrow \quad u, v$ in ancestor/descendant relation in F

$\mathrm{td}(G) :=$ least possible depth of an elimination forest of G.



Basic remarks:

– If G is connected, then F must be a tree.

– $H \subseteq G \quad \Rightarrow \quad \mathrm{td}(H) \leqslant \mathrm{td}(G)$

– $\mathrm{tw}(G) \leqslant \mathrm{td}(G) \leqslant \mathrm{tw}(G) \cdot \log n$

– $\mathrm{td}(G) = \mathrm{wcol}_\infty(G)$

Treedepth game

Treedepth game

Consider the following one-player **game** played on a graph G in rounds:

– **Each round:** Remove one vertex from each connected component.

Treedepth game

Consider the following one-player **game** played on a graph G in rounds:

– **Each round:** Remove one vertex from each connected component.

Fact. $\mathrm{td}(G) = \min$ # rounds needed to eliminate the whole graph

Treedepth game

Consider the following one-player **game** played on a graph G in rounds:

– **Each round:** Remove one vertex from each connected component.

Fact. $\mathrm{td}(G) = \min \#$ rounds needed to eliminate the whole graph

($\geqslant$): Eliminate elimination forest level by level.

Treedepth game

Consider the following one-player **game** played on a graph G in rounds:

– **Each round:** Remove one vertex from each connected component.

Fact. $\mathrm{td}(G) = \min$ # rounds needed to eliminate the whole graph

($\geqslant$): Eliminate elimination forest level by level.

($\leqslant$): Elimination strategy yields an elimination forest.

Treedepth game

Consider the following one-player **game** played on a graph G in rounds:

– **Each round:** Remove one vertex from each connected component.

Fact. $\mathrm{td}(G) =$ min # rounds needed to eliminate the whole graph

($\geqslant$): Eliminate elimination forest level by level.

($\leqslant$): Elimination strategy yields an elimination forest.

1. Compute:

$\mathrm{td}(K_n) =$

$\mathrm{td}(P_n) =$

Treedepth and bounded expansion

Intuition:

Treedepth and bounded expansion

Intuition:

– **Low td coloring:** Decomposition of a graph into **very simple** pieces.

Treedepth and bounded expansion

Intuition:

- **Low td coloring:** Decomposition of a graph into **very simple** pieces.
- **very simple** $\leftrightsquigarrow$ **bnd treedepth**

Treedepth and bounded expansion

Intuition:

- **Low td coloring:** Decomposition of a graph into **very simple** pieces.
- **very simple** $\leftrightsquigarrow$ **bnd treedepth**
- **Thm:** If $\mathcal{C}$ has **bnd expansion**,

  then each $G \in \mathcal{C}$ has such a decomposition.

Treedepth and bounded expansion

Intuition:

– **Low td coloring:** Decomposition of a graph into **very simple** pieces.

– **very simple** $\leftrightsquigarrow$ **bnd treedepth**

– **Thm:** If $\mathcal{C}$ has **bnd expansion**,

then each $G \in \mathcal{C}$ has such a decomposition.

Fix $\mathcal{C}$ of **bnd expansion**, $G \in \mathcal{C}$, and a parameter $p \in \mathbb{N}$.

Treedepth and bounded expansion

Intuition:

- **Low td coloring:** Decomposition of a graph into **very simple** pieces.
- **very simple** $\leftrightsquigarrow$ **bnd treedepth**
- **Thm:** If $\mathcal{C}$ has **bnd expansion**,

  then each $G \in \mathcal{C}$ has such a decomposition.

Fix $\mathcal{C}$ of **bnd expansion**, $G \in \mathcal{C}$, and a parameter $p \in \mathbb{N}$.

Let σ be a vertex ordering witnessing the value of $\mathrm{wcol}_{2p-1}(G)$.

$\longrightarrow$

Treedepth and bounded expansion

Intuition:

- **Low td coloring:** Decomposition of a graph into **very simple** pieces.
- **very simple** $\leftrightsquigarrow$ **bnd treedepth**
- **Thm:** If $\mathcal{C}$ has **bnd expansion**,

  then each $G \in \mathcal{C}$ has such a decomposition.

Fix $\mathcal{C}$ of **bnd expansion**, $G \in \mathcal{C}$, and a parameter $p \in \mathbb{N}$.

Let σ be a vertex ordering witnessing the value of $\mathrm{wcol}_{2p-1}(G)$.

Let ϕ be the greedy coloring with $\mathrm{wcol}_{2p-1}(G)$ colors s.t.:

For each v, $\phi(v) \notin$ colors given to $\mathrm{WReach}_{2p-1}(G) \setminus \{v\}$ by ϕ.

Constructing a low td coloring

1. P is a path on 2^{p-1} vertices $\quad\Rightarrow\quad$ P receives $\geqslant p$ different colors.

2. $H \subseteq G$ is connected and receives $\leqslant p$ colors $\quad\Rightarrow$

H has a vertex of unique color.

3. $H \subseteq G$ receives $\leqslant p$ colors $\quad\Rightarrow\quad$ $\mathrm{td}(H) \leqslant p$.

Low td colorings

Low td colorings

Theorem (low td colorings)

Let $\mathcal{C}$ be a class of **bnd expansion** and $p \in \mathbb{N}$. Then there is $M(p) \in \mathbb{N}$ such that every $G \in \mathcal{C}$ has a coloring with $M(p)$ colors satisfying:

Every p colors together induce a subgraph of treedepth $\leqslant p$.

Low td colorings

Theorem (low td colorings)

Let $\mathcal{C}$ be a class of **bnd expansion** and $p \in \mathbb{N}$. Then there is $M(p) \in \mathbb{N}$ such that every $G \in \mathcal{C}$ has a coloring with $M(p)$ colors satisfying:

Every p colors together induce a subgraph of treedepth $\leqslant p$.

Note: In our proof, $M(p) = \mathrm{wcol}_{2p-1}(\mathcal{C})$.

Low td colorings

Theorem (low td colorings)

Let $\mathcal{C}$ be a class of **bnd expansion** and $p \in \mathbb{N}$. Then there is $M(p) \in \mathbb{N}$ such that every $G \in \mathcal{C}$ has a coloring with $M(p)$ colors satisfying:

Every p colors together induce a subgraph of treedepth $\leqslant p$.

Note: In our proof, $M(p) = \mathrm{wcol}_{2p-1}(\mathcal{C})$.

Theorem (low td coverings)

Let $\mathcal{C}$ be a class of **bnd expansion** and $p \in \mathbb{N}$. Then there is $N(p) \in \mathbb{N}$ such that in every $G \in \mathcal{C}$ there are vertex subsets $A_1, \ldots, A_{N(p)}$ satisfying:

– $\mathrm{td}(G[A_i]) \leqslant p$ for each i; and

– for each $X \subseteq V(G)$ with $|X| \leqslant p$, there is A_i such that $X \subseteq A_i$.

Low td colorings

Theorem (low td colorings)

Let $\mathcal{C}$ be a class of **bnd expansion** and $p \in \mathbb{N}$. Then there is $M(p) \in \mathbb{N}$ such that every $G \in \mathcal{C}$ has a coloring with $M(p)$ colors satisfying:

Every p colors together induce a subgraph of treedepth $\leqslant p$.

Note: In our proof, $M(p) = \mathrm{wcol}_{2p-1}(\mathcal{C})$.

Theorem (low td coverings)

Let $\mathcal{C}$ be a class of **bnd expansion** and $p \in \mathbb{N}$. Then there is $N(p) \in \mathbb{N}$ such that in every $G \in \mathcal{C}$ there are vertex subsets $A_1, \ldots, A_{N(p)}$ satisfying:

– $\mathrm{td}(G[A_i]) \leqslant p$ for each i; and

– for each $X \subseteq V(G)$ with $|X| \leqslant p$, there is A_i such that $X \subseteq A_i$.

Proof:

Low td colorings

Theorem (low td colorings)

Let $\mathcal{C}$ be a class of **bnd expansion** and $p \in \mathbb{N}$. Then there is $M(p) \in \mathbb{N}$ such that every $G \in \mathcal{C}$ has a coloring with $M(p)$ colors satisfying:

Every p colors together induce a subgraph of treedepth $\leqslant p$.

Note: In our proof, $M(p) = \mathrm{wcol}_{2p-1}(\mathcal{C})$.

Theorem (low td coverings)

Let $\mathcal{C}$ be a class of **bnd expansion** and $p \in \mathbb{N}$. Then there is $N(p) \in \mathbb{N}$ such that in every $G \in \mathcal{C}$ there are vertex subsets $A_1, \ldots, A_{N(p)}$ satisfying:

- $\mathrm{td}(G[A_i]) \leqslant p$ for each i; and
- for each $X \subseteq V(G)$ with $|X| \leqslant p$, there is A_i such that $X \subseteq A_i$.

Proof:

- $\{A_1, A_2, \ldots, A_{N(p)}\} =$ subsets of p colors.

Low td colorings

Theorem (low td colorings)

Let $\mathcal{C}$ be a class of **bnd expansion** and $p \in \mathbb{N}$. Then there is $M(p) \in \mathbb{N}$ such that every $G \in \mathcal{C}$ has a coloring with $M(p)$ colors satisfying:

Every p colors together induce a subgraph of treedepth $\leqslant p$.

Note: In our proof, $M(p) = \mathrm{wcol}_{2p-1}(\mathcal{C})$.

Theorem (low td coverings)

Let $\mathcal{C}$ be a class of **bnd expansion** and $p \in \mathbb{N}$. Then there is $N(p) \in \mathbb{N}$ such that in every $G \in \mathcal{C}$ there are vertex subsets $A_1, \ldots, A_{N(p)}$ satisfying:

- $\mathrm{td}(G[A_i]) \leqslant p$ for each i; and
- for each $X \subseteq V(G)$ with $|X| \leqslant p$, there is A_i such that $X \subseteq A_i$.

Proof:

- $\{A_1, A_2, \ldots, A_{N(p)}\}$ = subsets of p colors.
- $N(p) = \binom{M(p)}{p}$

Algorithmic application

Algorithmic application

Subgraph Isomorphism:

For a fixed graph Q, check whether input G contains Q as a subgraph.

Algorithmic application

Subgraph Isomorphism:

For a fixed graph Q, check whether input G contains Q as a subgraph.

Let $p := |V(Q)|$. (Imagine $p = 50$)

Algorithmic application

Subgraph Isomorphism:

For a fixed graph Q, check whether input G contains Q as a subgraph.

Let $p := |V(Q)|$. (Imagine $p = 50$)

Trivial: running time $\mathcal{O}(|V(G)|^p)$.

Algorithmic application

Subgraph Isomorphism:

For a fixed graph Q, check whether input G contains Q as a subgraph.

Let $p \coloneqq |V(Q)|$. (Imagine $p = 50$)

Trivial: running time $\mathcal{O}(|V(G)|^p)$.

– In general, running time $|V(G)|^{o(p)}$ is unlikely.

Algorithmic application

Subgraph Isomorphism:

For a fixed graph Q, check whether input G contains Q as a subgraph.

Let $p \coloneqq |V(Q)|$. (Imagine $p = 50$)

Trivial: running time $\mathcal{O}(|V(G)|^p)$.

– In general, running time $|V(G)|^{o(p)}$ is unlikely.

Supposing $G \in \mathcal{C}$ where $\mathcal{C}$ has **bnd expansion**, we can do as follows:

Algorithmic application

Subgraph Isomorphism:

For a fixed graph Q, check whether input G contains Q as a subgraph.

Let $p := |V(Q)|$. (Imagine $p = 50$)

Trivial: running time $\mathcal{O}(|V(G)|^p)$.

– In general, running time $|V(G)|^{o(p)}$ is unlikely.

Supposing $G \in \mathcal{C}$ where $\mathcal{C}$ has **bnd expansion**, we can do as follows:

– Compute a treedepth-p cover $A_1, \ldots, A_{N(p)}$ of G.

Algorithmic application

Subgraph Isomorphism:

For a fixed graph Q, check whether input G contains Q as a subgraph.

Let $p := |V(Q)|$. (Imagine $p = 50$)

Trivial: running time $\mathcal{O}(|V(G)|^p)$.

– In general, running time $|V(G)|^{o(p)}$ is unlikely.

Supposing $G \in \mathcal{C}$ where $\mathcal{C}$ has **bnd expansion**, we can do as follows:

– Compute a treedepth-p cover $A_1, \ldots, A_{N(p)}$ of G.

– For each i, test if $Q \subseteq G[A_i]$ by dynamic programming in linear time.

Algorithmic application

Subgraph Isomorphism:

For a fixed graph Q, check whether input G contains Q as a subgraph.

Let $p := |V(Q)|$. (Imagine $p = 50$)

Trivial: running time $\mathcal{O}(|V(G)|^p)$.

– In general, running time $|V(G)|^{o(p)}$ is unlikely.

Supposing $G \in \mathcal{C}$ where $\mathcal{C}$ has **bnd expansion**, we can do as follows:

– Compute a treedepth-p cover $A_1, \ldots, A_{N(p)}$ of G.

– For each i, test if $Q \subseteq G[A_i]$ by dynamic programming in linear time.

– **Obs:** $Q \subseteq G \quad \Leftrightarrow \quad Q \subseteq G[A_i]$ for some i.

Algorithmic application

Subgraph Isomorphism:

For a fixed graph Q, check whether input G contains Q as a subgraph.

Let $p := |V(Q)|$. (Imagine $p = 50$)

Trivial: running time $\mathcal{O}(|V(G)|^p)$.

– In general, running time $|V(G)|^{o(p)}$ is unlikely.

Supposing $G \in \mathcal{C}$ where $\mathcal{C}$ has **bnd expansion**, we can do as follows:

– Compute a treedepth-p cover $A_1, \ldots, A_{N(p)}$ of G.

– For each i, test if $Q \subseteq G[A_i]$ by dynamic programming in linear time.

– **Obs:** $Q \subseteq G \quad \Leftrightarrow \quad Q \subseteq G[A_i]$ for some i.

Cor: A linear-time algorithm testing whether $Q \subseteq G$.

Combinatorial application

Combinatorial application

Theorem

Let $\mathcal{C}$ be a class of **bnd expansion** and $d \in \mathbb{N}$ be **odd**. Then there is $L(d) \in \mathbb{N}$ such that every $G \in \mathcal{C}$ has a coloring ϕ with $L(d)$ colors satisfying:

$$\mathrm{dist}(u, v) = d \qquad \Rightarrow \qquad \phi(u) \neq \phi(v).$$

Combinatorial application

Theorem

Let $\mathcal{C}$ be a class of **bnd expansion** and $d \in \mathbb{N}$ be **odd**. Then there is $L(d) \in \mathbb{N}$ such that every $G \in \mathcal{C}$ has a coloring ϕ with $L(d)$ colors satisfying:

$$\operatorname{dist}(u,v) = d \qquad \Rightarrow \qquad \phi(u) \neq \phi(v).$$

Lemma

If $\operatorname{td}(G) = p$, then G has a coloring with $2^p - 1$ colors such that

$$\operatorname{dist}(u,v) \text{ is } \textbf{odd} \qquad \Rightarrow \qquad \phi(u) \neq \phi(v).$$

Combinatorial application

Theorem

Let $\mathcal{C}$ be a class of **bnd expansion** and $d \in \mathbb{N}$ be **odd**. Then there is $L(d) \in \mathbb{N}$ such that every $G \in \mathcal{C}$ has a coloring ϕ with $L(d)$ colors satisfying:

$$\mathrm{dist}(u, v) = d \qquad \Rightarrow \qquad \phi(u) \neq \phi(v).$$

Lemma

If $\mathrm{td}(G) = p$, then G has a coloring with $2^p - 1$ colors such that

$$\mathrm{dist}(u, v) \text{ is } \textbf{odd} \qquad \Rightarrow \qquad \phi(u) \neq \phi(v).$$

Lemma $\Rightarrow$ **Theorem**:

Combinatorial application

Theorem

Let $\mathcal{C}$ be a class of **bnd expansion** and $d \in \mathbb{N}$ be **odd**. Then there is $L(d) \in \mathbb{N}$ such that every $G \in \mathcal{C}$ has a coloring ϕ with $L(d)$ colors satisfying:

$$\mathrm{dist}(u, v) = d \qquad \Rightarrow \qquad \phi(u) \neq \phi(v).$$

Lemma

If $\mathrm{td}(G) = p$, then G has a coloring with $2^p - 1$ colors such that

$$\mathrm{dist}(u, v) \text{ is } \textbf{odd} \qquad \Rightarrow \qquad \phi(u) \neq \phi(v).$$

Lemma $\Rightarrow$ **Theorem**:

– Let $\{A_1, \ldots, A_{N(d+1)}\}$ be a treedepth-$(d+1)$ covering of G.

Combinatorial application

Theorem

Let $\mathcal{C}$ be a class of **bnd expansion** and $d \in \mathbb{N}$ be **odd**. Then there is $L(d) \in \mathbb{N}$ such that every $G \in \mathcal{C}$ has a coloring ϕ with $L(d)$ colors satisfying:

$$\mathrm{dist}(u, v) = d \qquad \Rightarrow \qquad \phi(u) \neq \phi(v).$$

Lemma

If $\mathrm{td}(G) = p$, then G has a coloring with $2^p - 1$ colors such that

$$\mathrm{dist}(u, v) \text{ is } \textbf{odd} \qquad \Rightarrow \qquad \phi(u) \neq \phi(v).$$

Lemma $\Rightarrow$ **Theorem**:

- Let $\{A_1, \ldots, A_{N(d+1)}\}$ be a treedepth-$(d+1)$ covering of G.
- Color each $G[A_i]$ using **Lemma** with $2^{d+1} - 1$ colors.

Combinatorial application

Theorem

Let $\mathcal{C}$ be a class of **bnd expansion** and $d \in \mathbb{N}$ be **odd**. Then there is $L(d) \in \mathbb{N}$ such that every $G \in \mathcal{C}$ has a coloring ϕ with $L(d)$ colors satisfying:

$$\mathrm{dist}(u, v) = d \qquad \Rightarrow \qquad \phi(u) \neq \phi(v).$$

Lemma

If $\mathrm{td}(G) = p$, then G has a coloring with $2^p - 1$ colors such that

$$\mathrm{dist}(u, v) \text{ is } \textbf{odd} \qquad \Rightarrow \qquad \phi(u) \neq \phi(v).$$

Lemma $\Rightarrow$ **Theorem**:

- Let $\{A_1, \ldots, A_{N(d+1)}\}$ be a treedepth-$(d + 1)$ covering of G.
- Color each $G[A_i]$ using **Lemma** with $2^{d+1} - 1$ colors.
- Construct the product coloring.

Combinatorial application

Theorem

Let $\mathcal{C}$ be a class of **bnd expansion** and $d \in \mathbb{N}$ be **odd**. Then there is $L(d) \in \mathbb{N}$ such that every $G \in \mathcal{C}$ has a coloring ϕ with $L(d)$ colors satisfying:

$$\mathrm{dist}(u, v) = d \qquad \Rightarrow \qquad \phi(u) \neq \phi(v).$$

Lemma

If $\mathrm{td}(G) = p$, then G has a coloring with $2^p - 1$ colors such that

$$\mathrm{dist}(u, v) \text{ is } \textbf{odd} \qquad \Rightarrow \qquad \phi(u) \neq \phi(v).$$

Lemma $\Rightarrow$ **Theorem**:

- Let $\{A_1, \ldots, A_{N(d+1)}\}$ be a treedepth-$(d+1)$ covering of G.
- Color each $G[A_i]$ using **Lemma** with $2^{d+1} - 1$ colors.
- Construct the product coloring.
- $L(d) = \left(2^{d+1}\right)^{N(d+1)}$.

Combinatorial application

Theorem

Let $\mathcal{C}$ be a class of **bnd expansion** and $d \in \mathbb{N}$ be **odd**. Then there is $L(d) \in \mathbb{N}$ such that every $G \in \mathcal{C}$ has a coloring ϕ with $L(d)$ colors satisfying:

$$\mathrm{dist}(u, v) = d \qquad \Rightarrow \qquad \phi(u) \neq \phi(v).$$

Lemma

If $\mathrm{td}(G) = p$, then G has a coloring with $2^p - 1$ colors such that

$$\mathrm{dist}(u, v) \text{ is } \textbf{odd} \qquad \Rightarrow \qquad \phi(u) \neq \phi(v).$$

Lemma $\Rightarrow$ **Theorem**:

- Let $\{A_1, \ldots, A_{N(d+1)}\}$ be a treedepth-$(d+1)$ covering of G.
- Color each $G[A_i]$ using **Lemma** with $2^{d+1} - 1$ colors.
- Construct the product coloring.
- $L(d) = \left(2^{d+1}\right)^{N(d+1)}$.
- **Obs:** A u-to-v path of length d is entirely contained in some $G[A_i]$.

Structural measures: summary

Structural measures: summary

Generalized coloring numbers:

Structural measures: summary

Generalized coloring numbers:

– Generalizations of degeneracy to connections of bounded length.

Structural measures: summary

Generalized coloring numbers:

- Generalizations of degeneracy to connections of bounded length.
- **Intuition:** $\mathrm{WReach}_d[G, \sigma, v]$ **guards** short connections from v.

Structural measures: summary

Generalized coloring numbers:

- Generalizations of degeneracy to connections of bounded length.
- **Intuition:** $\mathrm{WReach}_d[G, \sigma, v]$ **guards** short connections from v.
- **Main trick:** Consider the σ-smallest vertex.

Structural measures: summary

Generalized coloring numbers:

- Generalizations of degeneracy to connections of bounded length.
- **Intuition:** $\mathrm{WReach}_d[G, \sigma, v]$ **guards** short connections from v.
- **Main trick:** Consider the σ-smallest vertex.

Low treedepth coverings:

Structural measures: summary

Generalized coloring numbers:

- Generalizations of degeneracy to connections of bounded length.
- **Intuition:** $\mathrm{WReach}_d[G, \sigma, v]$ **guards** short connections from v.
- **Main trick:** Consider the σ-smallest vertex.

Low treedepth coverings:

- **Global decomposition** into simple pieces.

Structural measures: summary

Generalized coloring numbers:

- Generalizations of degeneracy to connections of bounded length.
- **Intuition:** $\mathrm{WReach}_d[G, \sigma, v]$ **guards** short connections from v.
- **Main trick:** Consider the σ-smallest vertex.

Low treedepth coverings:

- **Global decomposition** into simple pieces.
- Provide a **reduction scheme**:

bnd treedepth $\Rightarrow$ **bnd expansion**

Structural measures: summary

Generalized coloring numbers:

– Generalizations of degeneracy to connections of bounded length.

– **Intuition:** $\mathrm{WReach}_d[G, \sigma, v]$ **guards** short connections from v.

– **Main trick:** Consider the σ-smallest vertex.

Low treedepth coverings:

– **Global decomposition** into simple pieces.

– Provide a **reduction scheme**:

$$\textbf{bnd treedepth} \qquad \Rightarrow \qquad \textbf{bnd expansion}$$

These are main tools for **bounded expansion** classes.

Structural measures: summary

Generalized coloring numbers:

- Generalizations of degeneracy to connections of bounded length.
- **Intuition:** $\mathrm{WReach}_d[G, \sigma, v]$ **guards** short connections from v.
- **Main trick:** Consider the σ-smallest vertex.

Low treedepth coverings:

- **Global decomposition** into simple pieces.
- Provide a **reduction scheme**:

$$\textbf{bnd treedepth} \quad \Rightarrow \quad \textbf{bnd expansion}$$

These are main tools for **bounded expansion** classes.

In **nowhere dense** classes they also work, but:

bounded by a constant $\quad \rightsquigarrow \quad$ bounded by $\mathcal{O}(n^{\varepsilon})$ for any $\varepsilon > 0$

Structural measures: summary

Generalized coloring numbers:

– Generalizations of degeneracy to connections of bounded length.

– **Intuition:** $\mathrm{WReach}_d[G, \sigma, v]$ **guards** short connections from v.

– **Main trick:** Consider the σ-smallest vertex.

Low treedepth coverings:

– **Global decomposition** into simple pieces.

– Provide a **reduction scheme**:

bnd treedepth $\Rightarrow$ **bnd expansion**

These are main tools for **bounded expansion** classes.

In **nowhere dense** classes they also work, but:

bounded by a constant $\rightsquigarrow$ bounded by $\mathcal{O}(n^{\varepsilon})$ for any $\varepsilon > 0$

Many arguments become much more technical or completely fail.

Structural measures: summary

Generalized coloring numbers:

- Generalizations of degeneracy to connections of bounded length.
- **Intuition:** $\mathrm{WReach}_d[G, \sigma, v]$ **guards** short connections from v.
- **Main trick:** Consider the σ-smallest vertex.

Low treedepth coverings:

- **Global decomposition** into simple pieces.
- Provide a **reduction scheme**:

$$\textbf{bnd treedepth} \quad \Rightarrow \quad \textbf{bnd expansion}$$

These are main tools for **bounded expansion** classes.

In **nowhere dense** classes they also work, but:

$$\text{bounded by a constant} \quad \rightsquigarrow \quad \text{bounded by } \mathcal{O}(n^{\varepsilon}) \text{ for any } \varepsilon > 0$$

Many arguments become much more technical or completely fail.

Main tool for **nowhere dense** classes: **uniform quasi-wideness**.

Part 4:

Uniform quasi-wideness and ladders

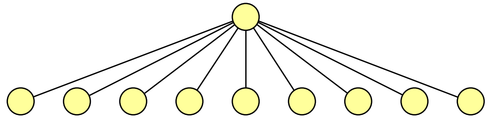
Wideness in graphs

Int: In a **huge** sparse graph, there are **many** vertices that are pairwise **far** from each other.

Wideness in graphs

Int: In a **huge** sparse graph, there are **many** vertices that are pairwise **far** from each other.

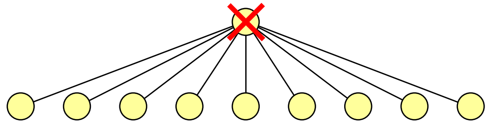
Wrong: a star.



Wideness in graphs

Int: In a **huge** sparse graph, there are **many** vertices that are pairwise **far** from each other.

Wrong: a star.



Wideness in graphs

Wideness in graphs

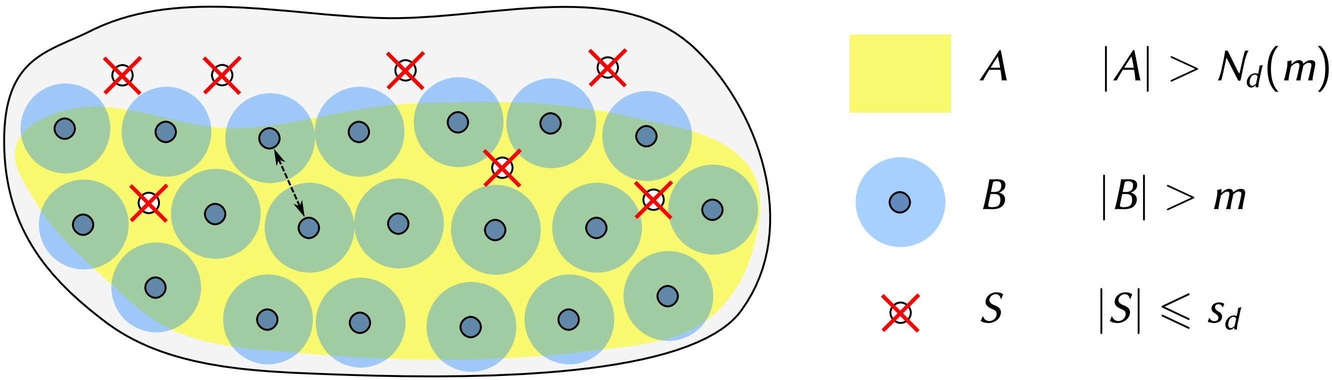
Int: In a **huge** sparse graph, one can remove **few** vertices so that there are **many** vertices that are pairwise **far** from each other.

Wideness in graphs

Int: In a **huge** sparse graph, one can remove **few** vertices so that there are **many** vertices that are pairwise **far** from each other.

Definition (Uniform quasi-wideness)

A class $\mathcal{C}$ is **uqw** if for every $d \in \mathbb{N}$ there exist $s_d \in \mathbb{N}$ and $N_d \colon \mathbb{N} \to \mathbb{N}$ s.t.

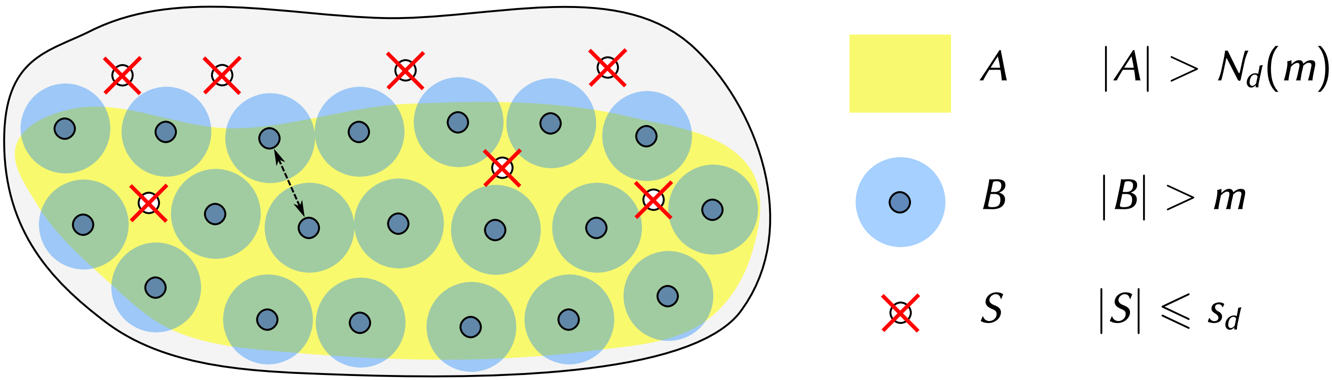


Wideness in graphs

Int: In a **huge** sparse graph, one can remove **few** vertices so that there are **many** vertices that are pairwise **far** from each other.

Definition (Uniform quasi-wideness)

A class $\mathcal{C}$ is **uqw** if for every $d \in \mathbb{N}$ there exist $s_d \in \mathbb{N}$ and $N_d \colon \mathbb{N} \to \mathbb{N}$ s.t. for every $G \in \mathcal{C}$, $m \in \mathbb{N}$, and $A \subseteq V(G)$ satisfying $|A| > N_d(m)$,

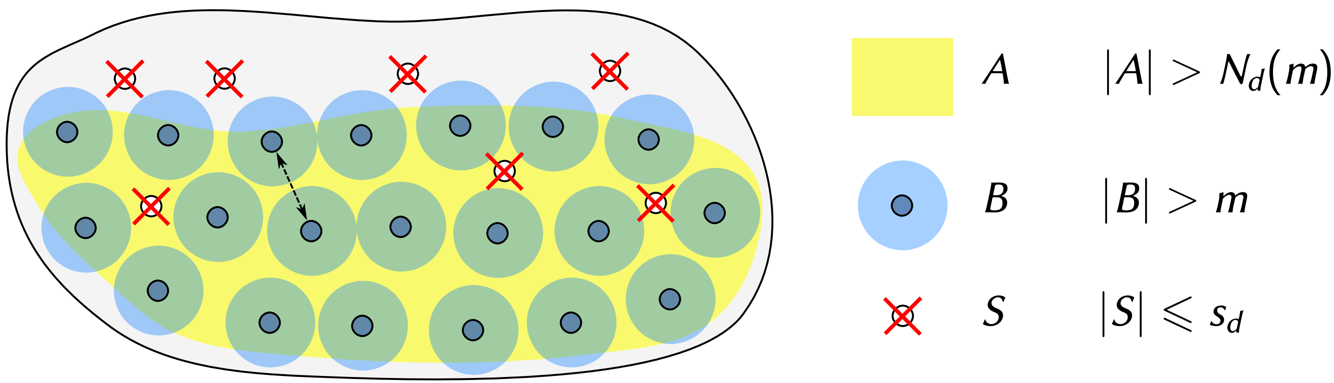


Wideness in graphs

Int: In a **huge** sparse graph, one can remove **few** vertices so that there are **many** vertices that are pairwise **far** from each other.

Definition (Uniform quasi-wideness)

A class $\mathcal{C}$ is **uqw** if for every $d \in \mathbb{N}$ there exist $s_d \in \mathbb{N}$ and $N_d \colon \mathbb{N} \to \mathbb{N}$ s.t. for every $G \in \mathcal{C}$, $m \in \mathbb{N}$, and $A \subseteq V(G)$ satisfying $|A| > N_d(m)$, there exists $S \subseteq V(G)$ and $B \subseteq A - S$ such that



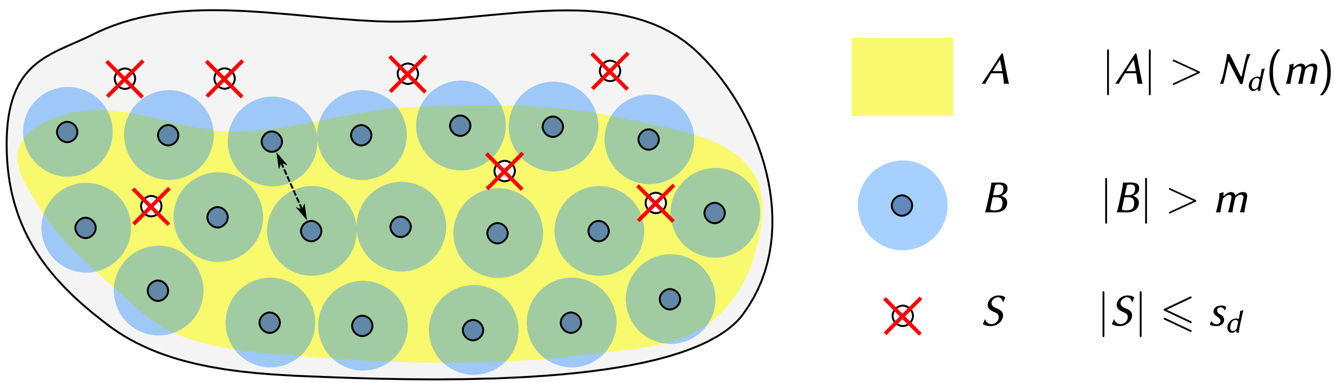
Wideness in graphs

Int: In a **huge** sparse graph, one can remove **few** vertices so that there are **many** vertices that are pairwise **far** from each other.

Definition (Uniform quasi-wideness)

A class $\mathcal{C}$ is **uqw** if for every $d \in \mathbb{N}$ there exist $s_d \in \mathbb{N}$ and $N_d \colon \mathbb{N} \to \mathbb{N}$ s.t. for every $G \in \mathcal{C}$, $m \in \mathbb{N}$, and $A \subseteq V(G)$ satisfying $|A| > N_d(m)$, there exists $S \subseteq V(G)$ and $B \subseteq A - S$ such that

- $|S| \leqslant s_d$; and



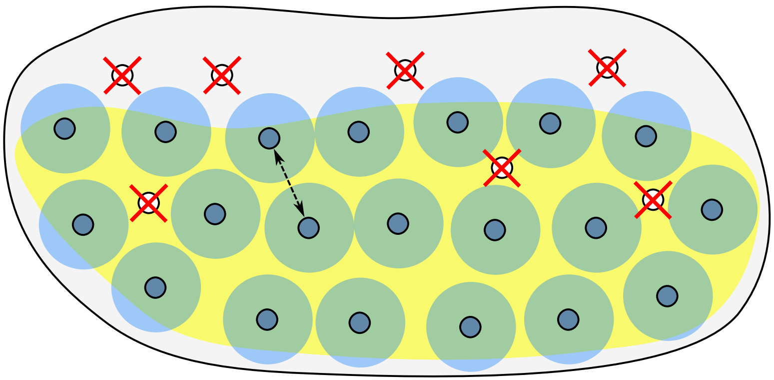
Wideness in graphs

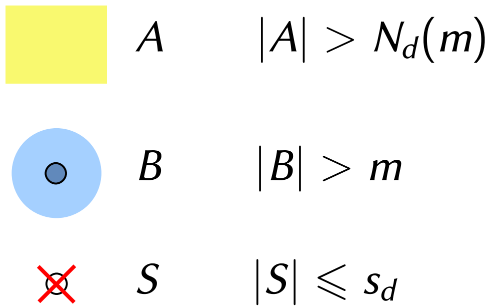
Int: In a **huge** sparse graph, one can remove **few** vertices so that there are **many** vertices that are pairwise **far** from each other.

Definition (Uniform quasi-wideness)

A class $\mathcal{C}$ is **uqw** if for every $d \in \mathbb{N}$ there exist $s_d \in \mathbb{N}$ and $N_d \colon \mathbb{N} \to \mathbb{N}$ s.t. for every $G \in \mathcal{C}$, $m \in \mathbb{N}$, and $A \subseteq V(G)$ satisfying $|A| > N_d(m)$, there exists $S \subseteq V(G)$ and $B \subseteq A - S$ such that

- $|S| \leqslant s_d$; and
- $|B| > m$ and $\mathrm{dist}_{G-S}(u, v) > d$ for all distinct $u, v \in B$.





Nowhere denseness and uqw

Theorem (Nešetřil and Ossona de Mendez)

A class of graphs is **nowhere dense** **iff** it is **uniformly quasi-wide**.

Nowhere denseness and uqw

Theorem (Nešetřil and Ossona de Mendez)

A class of graphs is **nowhere dense** **iff** it is **uniformly quasi-wide**.

Note: One can always have $N_d(m) = \text{poly}(m)$.

Nowhere denseness and uqw

Theorem (Nešetřil and Ossona de Mendez)

A class of graphs is **nowhere dense** **iff** it is **uniformly quasi-wide**.

Note: One can always have $N_d(m) = \mathrm{poly}(m)$.

Uniform quasi-wideness is a **Ramseyan** tool for **digging structure**.

Nowhere denseness and uqw

Theorem (Nešetřil and Ossona de Mendez)

A class of graphs is **nowhere dense** **iff** it is **uniformly quasi-wide**.

Note: One can always have $N_d(m) = \mathrm{poly}(m)$.

Uniform quasi-wideness is a **Ramseyan** tool for **digging structure**.

- Suppose in a graph G we have some **huge** complicated structure.

Nowhere denseness and uqw

Theorem (Nešetřil and Ossona de Mendez)

A class of graphs is **nowhere dense** **iff** it is **uniformly quasi-wide**.

Note: One can always have $N_d(m) = \text{poly}(m)$.

Uniform quasi-wideness is a **Ramseyan** tool for **digging structure**.

- Suppose in a graph G we have some **huge** complicated structure.
- Hit it with **uqw** to get a **large** well-behaved structure.

Nowhere denseness and uqw

Theorem (Nešetřil and Ossona de Mendez)

A class of graphs is **nowhere dense** **iff** it is **uniformly quasi-wide**.

Note: One can always have $N_d(m) = \mathrm{poly}(m)$.

Uniform quasi-wideness is a **Ramseyan** tool for **digging structure**.

- Suppose in a graph G we have some **huge** complicated structure.
- Hit it with **uqw** to get a **large** well-behaved structure.

Now: Application to **distance-d domination**.

Nowhere denseness and uqw

Theorem (Nešetřil and Ossona de Mendez)

A class of graphs is **nowhere dense** **iff** it is **uniformly quasi-wide**.

Note: One can always have $N_d(m) = \mathrm{poly}(m)$.

Uniform quasi-wideness is a **Ramseyan** tool for **digging structure**.

– Suppose in a graph G we have some **huge** complicated structure.

– Hit it with **uqw** to get a **large** well-behaved structure.

Now: Application to **distance-d domination**.

– Let $\mathcal{C}$ be nowhere dense and $d \in \mathbb{N}$ be fixed.

Nowhere denseness and uqw

Theorem (Nešetřil and Ossona de Mendez)

A class of graphs is **nowhere dense** **iff** it is **uniformly quasi-wide**.

Note: One can always have $N_d(m) = \mathrm{poly}(m)$.

Uniform quasi-wideness is a **Ramseyan** tool for **digging structure**.

– Suppose in a graph G we have some **huge** complicated structure.

– Hit it with **uqw** to get a **large** well-behaved structure.

Now: Application to **distance-d domination**.

– Let $\mathcal{C}$ be nowhere dense and $d \in \mathbb{N}$ be fixed.

– Given $G \in \mathcal{C}$ and $k \in \mathbb{N}$, decide whether $\mathrm{dom}_d(G) \leqslant k$.

Nowhere denseness and uqw

Theorem (Nešetřil and Ossona de Mendez)

A class of graphs is **nowhere dense** **iff** it is **uniformly quasi-wide**.

Note: One can always have $N_d(m) = \mathrm{poly}(m)$.

Uniform quasi-wideness is a **Ramseyan** tool for **digging structure**.

- Suppose in a graph G we have some **huge** complicated structure.
- Hit it with **uqw** to get a **large** well-behaved structure.

Now: Application to **distance-d domination**.

- Let $\mathcal{C}$ be nowhere dense and $d \in \mathbb{N}$ be fixed.
- Given $G \in \mathcal{C}$ and $k \in \mathbb{N}$, decide whether $\mathrm{dom}_d(G) \leqslant k$.
- **Trivial:** $\mathcal{O}(|V(G)|^k)$. We want something faster.

Nowhere denseness and uqw

Theorem (Nešetřil and Ossona de Mendez)

A class of graphs is **nowhere dense** **iff** it is **uniformly quasi-wide**.

Note: One can always have $N_d(m) = \text{poly}(m)$.

Uniform quasi-wideness is a **Ramseyan** tool for **digging structure**.

– Suppose in a graph G we have some **huge** complicated structure.

– Hit it with **uqw** to get a **large** well-behaved structure.

Now: Application to **distance-d domination**.

– Let $\mathcal{C}$ be nowhere dense and $d \in \mathbb{N}$ be fixed.

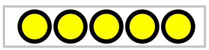
– Given $G \in \mathcal{C}$ and $k \in \mathbb{N}$, decide whether $\text{dom}_d(G) \leqslant k$.

– **Trivial:** $\mathcal{O}(|V(G)|^k)$. We want something faster.

Example from the work with Fabiański, Siebertz, and Toruńczyk.

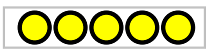
Algorithm

Algorithm

D_1 

Round 1: Take any k-tuple of vertices D_1.

Algorithm

D_1 

Round 1: Take any k-tuple of vertices D_1.

If D_1 is a dist-d domset, terminate.

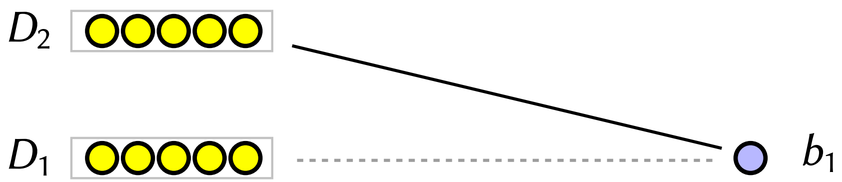
Algorithm

Round 1: Take any k-tuple of vertices D_1.

If D_1 is a dist-d domset, terminate.

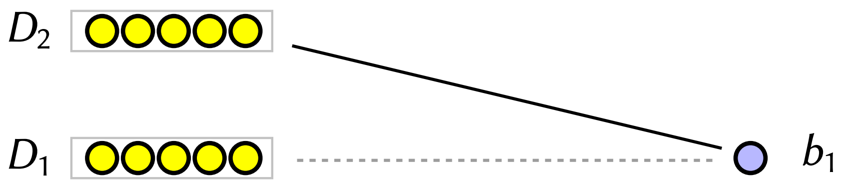
Otherwise there is an undominated vertex b_1.

Algorithm



Round 2: Find any k-tuple of vertices D_2 that dist-d dominates b_1.

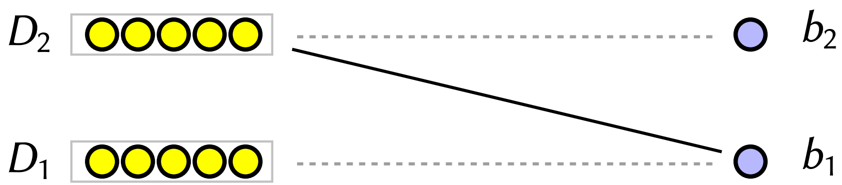
Algorithm



Round 2: Find any k-tuple of vertices D_2 that dist-d dominates b_1.

If D_2 is a dist-d domset, terminate.

Algorithm

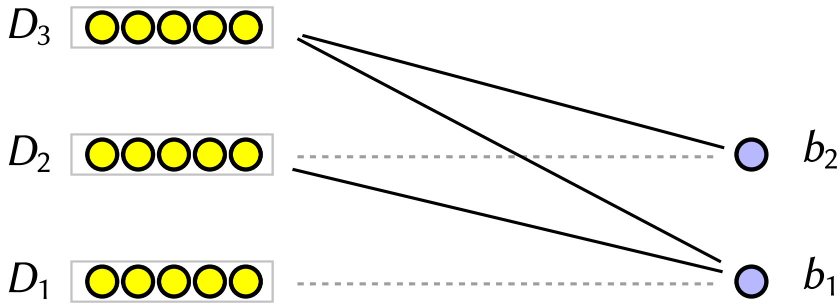


Round 2: Find any k-tuple of vertices D_2 that dist-d dominates b_1.

If D_2 is a dist-d domset, terminate.

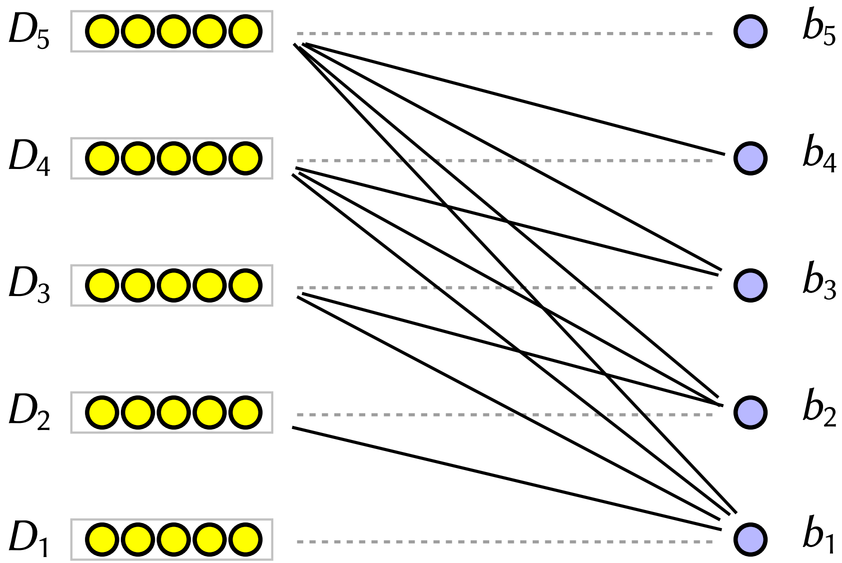
Otherwise there is an undominated vertex b_2.

Algorithm



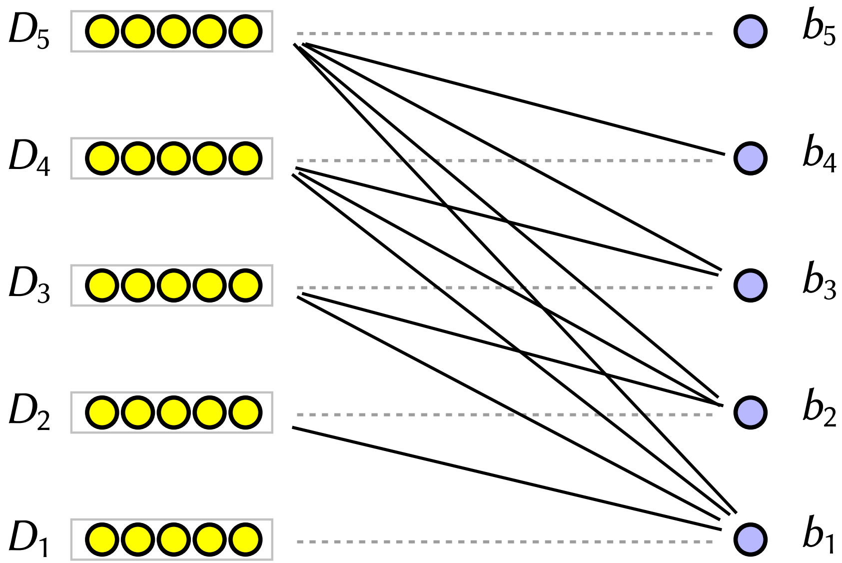
Round 3: Find any k-tuple of vertices D_3 that dist-d dominates $\{b_1, b_2\}$.

Algorithm



Round i: Find any k-tuple of vertices D_i that dominates $\{b_1, \ldots, b_{i-1}\}$.

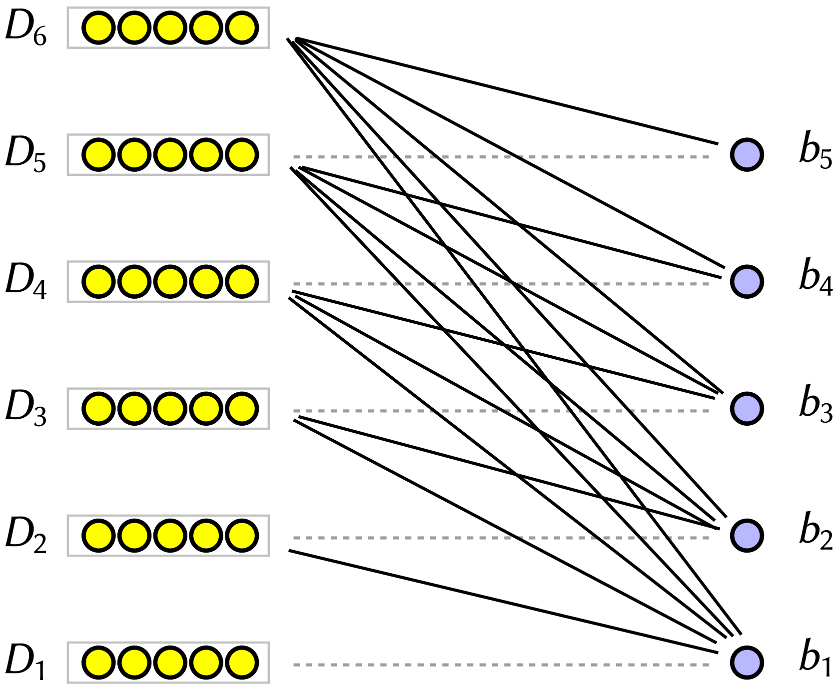
Algorithm



Round i: Find any k-tuple of vertices D_i that dominates $\{b_1, \ldots, b_{i-1}\}$.

If there is none, answer **NO**.

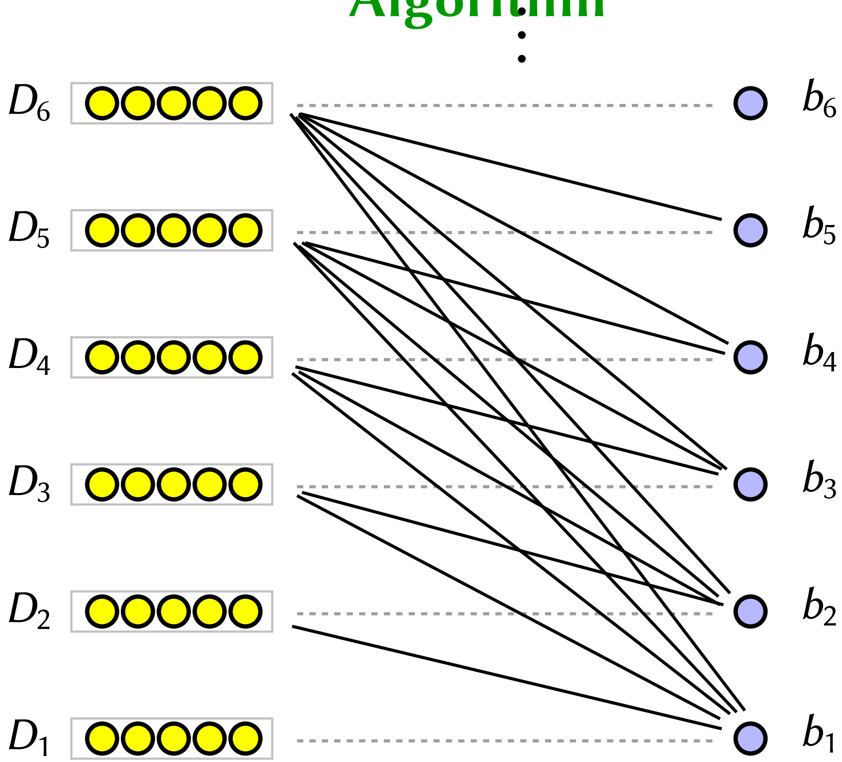
Algorithm



Round i: Find any k-tuple of vertices D_i that dominates $\{b_1, \ldots, b_{i-1}\}$.

If there is none, answer **NO**. If D_i is a dist-d domset, answer **YES**.

Algorithm



Round i: Find any k-tuple of vertices D_i that dominates $\{b_1, \ldots, b_{i-1}\}$.

If there is none, answer **NO**. If D_i is a dist-d domset, answer **YES**.

Otherwise there is an undominated vertex b_i. **Proceed**

Analysis

Analysis

The *i*th iteration can be performed in time $f(k, i) \cdot \|G\|$.

Analysis

The *i*th iteration can be performed in time $f(k, i) \cdot \|G\|$.

Why the number of iterations should be bounded?

Analysis

The ith iteration can be performed in time $f(k, i) \cdot \|G\|$.

Why the number of iterations should be bounded?

Lemma

The Algorithm terminates after at most $c \cdot k^c$ rounds, where c is a constant that depends only on $\mathcal{C}$ and d.

Analysis

The ith iteration can be performed in time $f(k, i) \cdot \|G\|$.

Why the number of iterations should be bounded?

Lemma

The Algorithm terminates after at most $c \cdot k^c$ rounds, where c is a constant that depends only on $\mathcal{C}$ and d.

Cor: A linear-time fixed-parameter algorithm.

Analysis

The ith iteration can be performed in time $f(k, i) \cdot \|G\|$.

Why the number of iterations should be bounded?

Lemma

The Algorithm terminates after at most $c \cdot k^c$ rounds, where c is a constant that depends only on $\mathcal{C}$ and d.

Cor: A linear-time fixed-parameter algorithm.

– Running time $f(k) \cdot \|G\|$, where f depends on $\mathcal{C}$ and d.

Analysis

The ith iteration can be performed in time $f(k, i) \cdot \|G\|$.

Why the number of iterations should be bounded?

Lemma

The Algorithm terminates after at most $c \cdot k^c$ rounds, where c is a constant that depends only on $\mathcal{C}$ and d.

Cor: A linear-time fixed-parameter algorithm.

– Running time $f(k) \cdot \|G\|$, where f depends on $\mathcal{C}$ and d.

Intuition:

Analysis

The ith iteration can be performed in time $f(k,i) \cdot \|G\|$.

Why the number of iterations should be bounded?

Lemma

The Algorithm terminates after at most $c \cdot k^c$ rounds, where c is a constant that depends only on $\mathcal{C}$ and d.

Cor: A linear-time fixed-parameter algorithm.

– Running time $f(k) \cdot \|G\|$, where f depends on $\mathcal{C}$ and d.

Intuition:

– The algorithms gradually gathers difficult witnesses.

Analysis

The ith iteration can be performed in time $f(k,i) \cdot \|G\|$.

Why the number of iterations should be bounded?

Lemma

The Algorithm terminates after at most $c \cdot k^c$ rounds, where c is a constant that depends only on $\mathcal{C}$ and d.

Cor: A linear-time fixed-parameter algorithm.

– Running time $f(k) \cdot \|G\|$, where f depends on $\mathcal{C}$ and d.

Intuition:

– The algorithms gradually gathers difficult witnesses.

– Eventually, domination of the witnesses forces domination of G.

Analysis

The ith iteration can be performed in time $f(k,i) \cdot \|G\|$.

Why the number of iterations should be bounded?

Lemma

The Algorithm terminates after at most $c \cdot k^c$ rounds, where c is a constant that depends only on $\mathcal{C}$ and d.

Cor: A linear-time fixed-parameter algorithm.

– Running time $f(k) \cdot \|G\|$, where f depends on $\mathcal{C}$ and d.

Intuition:

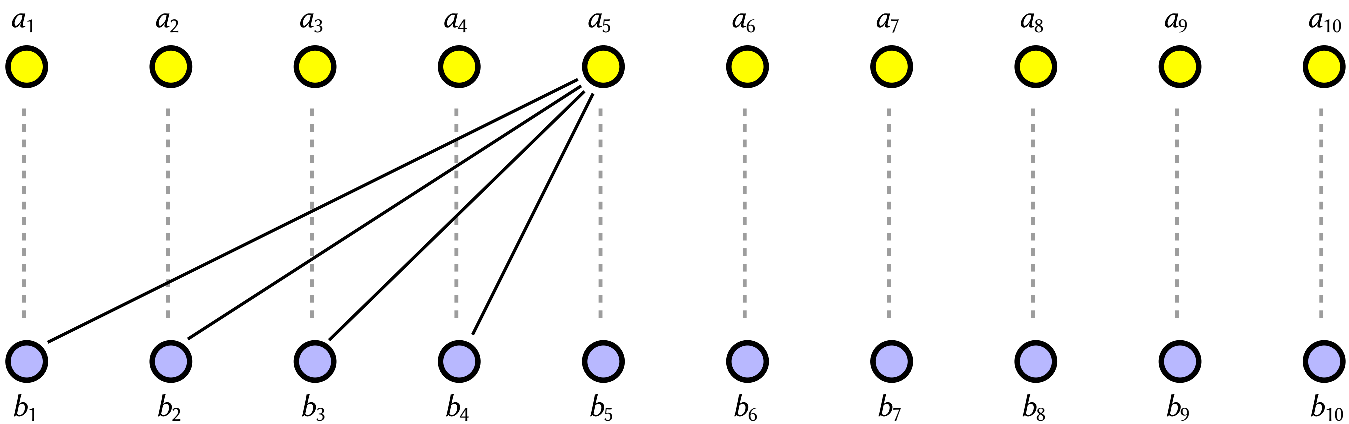
– The algorithms gradually gathers difficult witnesses.

– Eventually, domination of the witnesses forces domination of G.

Now: Proof of the **Lemma** for $k = 1$.

Semi-ladders

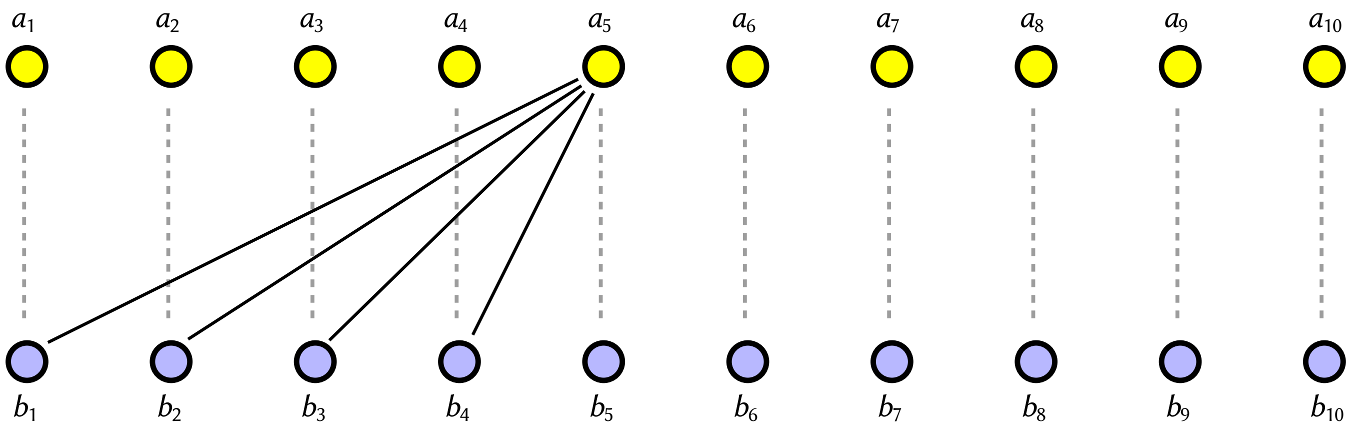
After ℓ rounds, the Algorithm has constructed a **semi-ladder** of order ℓ.



Semi-ladders

After ℓ rounds, the Algorithm has constructed a **semi-ladder** of order ℓ.

- Two sequences of vertices: $a_1, \ldots, a_\ell$ and $b_1, \ldots, b_\ell$.

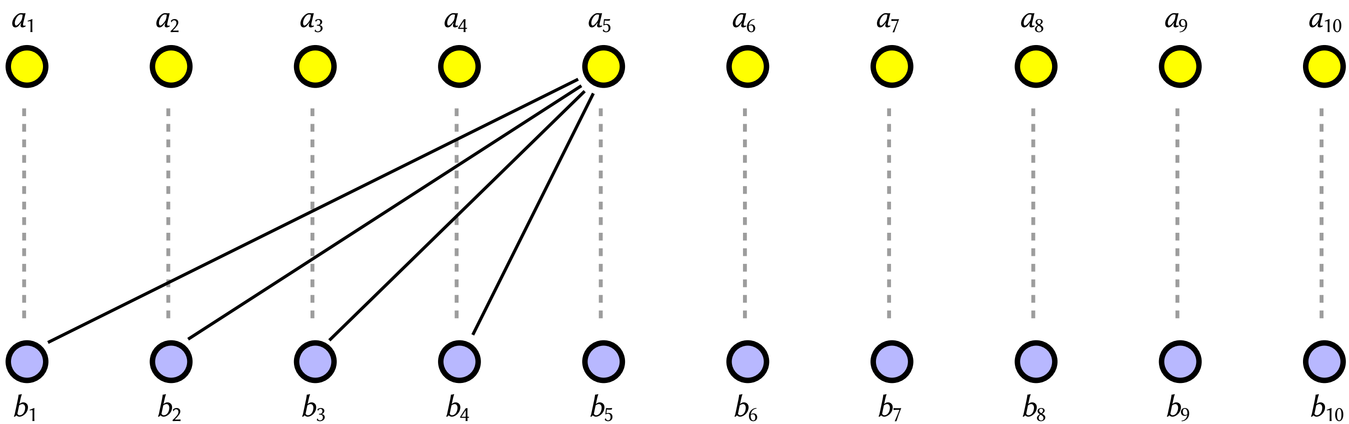


Semi-ladders

After ℓ rounds, the Algorithm has constructed a **semi-ladder** of order ℓ.

– Two sequences of vertices: $a_1, \ldots, a_\ell$ and $b_1, \ldots, b_\ell$.

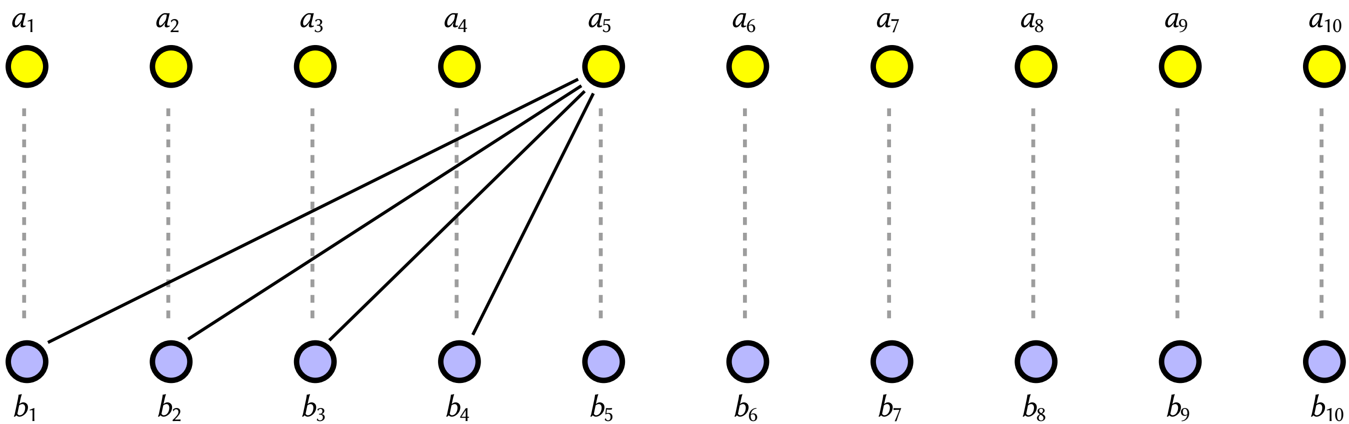
– For each i, we have $\text{dist}(a_i, b_i) > d$.



Semi-ladders

After ℓ rounds, the Algorithm has constructed a **semi-ladder** of order ℓ.

- Two sequences of vertices: $a_1, \ldots, a_\ell$ and $b_1, \ldots, b_\ell$.
- For each i, we have $\mathrm{dist}(a_i, b_i) > d$.
- For each $j < i$, we have $\mathrm{dist}(a_i, b_j) \leqslant d$.



Semi-ladders

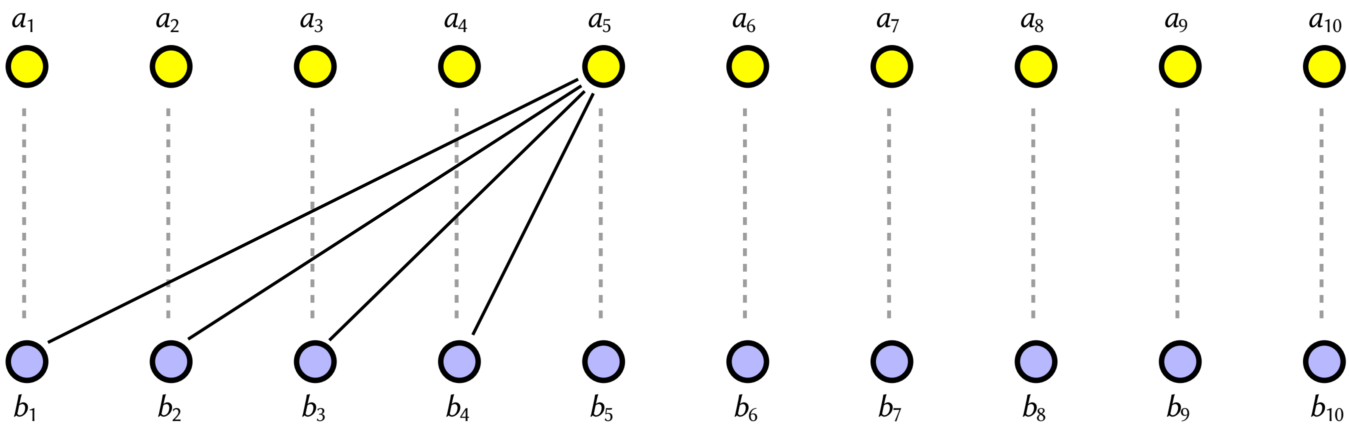
After ℓ rounds, the Algorithm has constructed a **semi-ladder** of order ℓ.

– Two sequences of vertices: $a_1, \ldots, a_\ell$ and $b_1, \ldots, b_\ell$.

– For each i, we have $\mathrm{dist}(a_i, b_i) > d$.

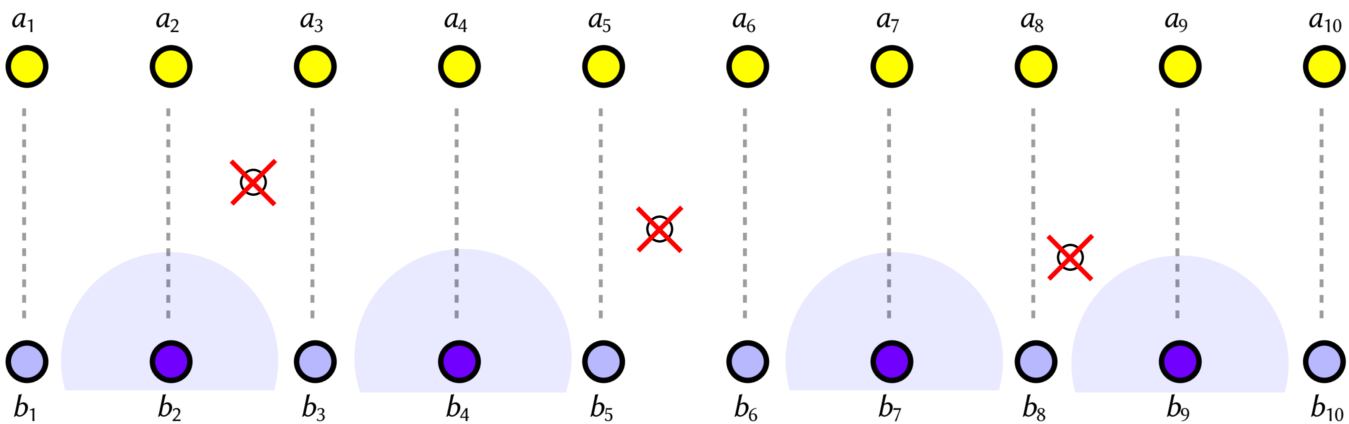
– For each $j < i$, we have $\mathrm{dist}(a_i, b_j) \leqslant d$.

Suppose $\ell > N(2(d+1)^s)$, where $N(\cdot) = N_{2d}(\cdot)$ and $s \coloneqq s_{2d}$.



Semi-ladders

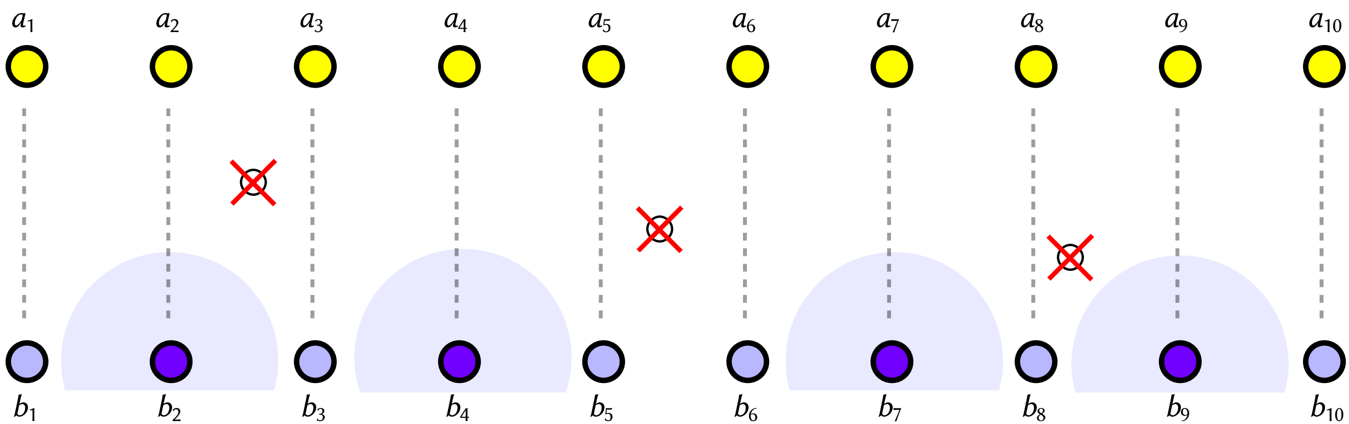
From **uqw** we get:



Semi-ladders

From **uqw** we get:

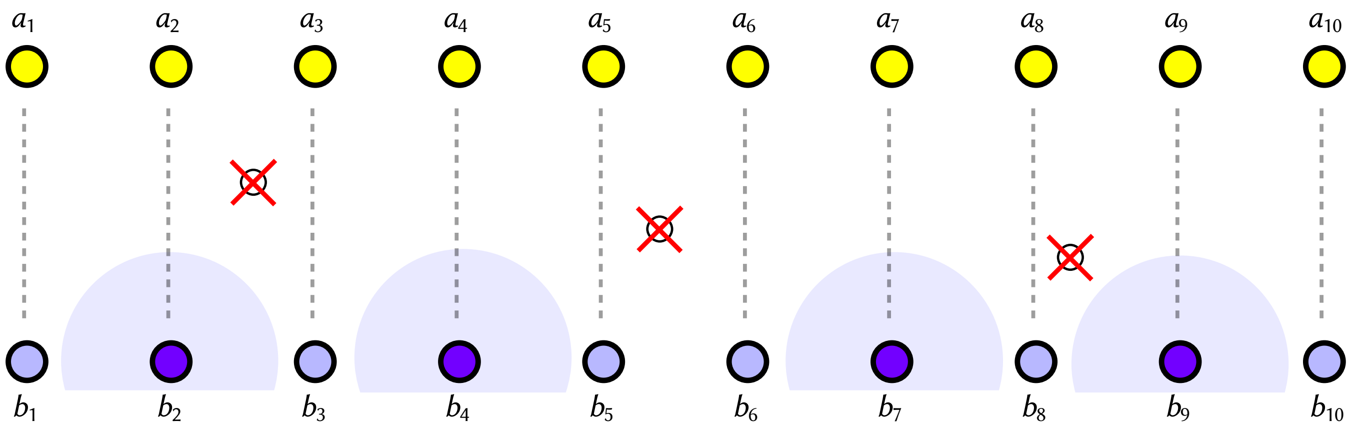
- set S satisfying $|S| \leqslant s$; and



Semi-ladders

From **uqw** we get:

- set S satisfying $|S| \leqslant s$; and
- $B \subseteq \{b_1, \ldots, b_\ell\}$ s.t. $|B| > 2(d+1)^s$ and $\mathrm{dist}_{G-S}(b_i, b_j) > 2d$ for $b_i, b_j \in B$.



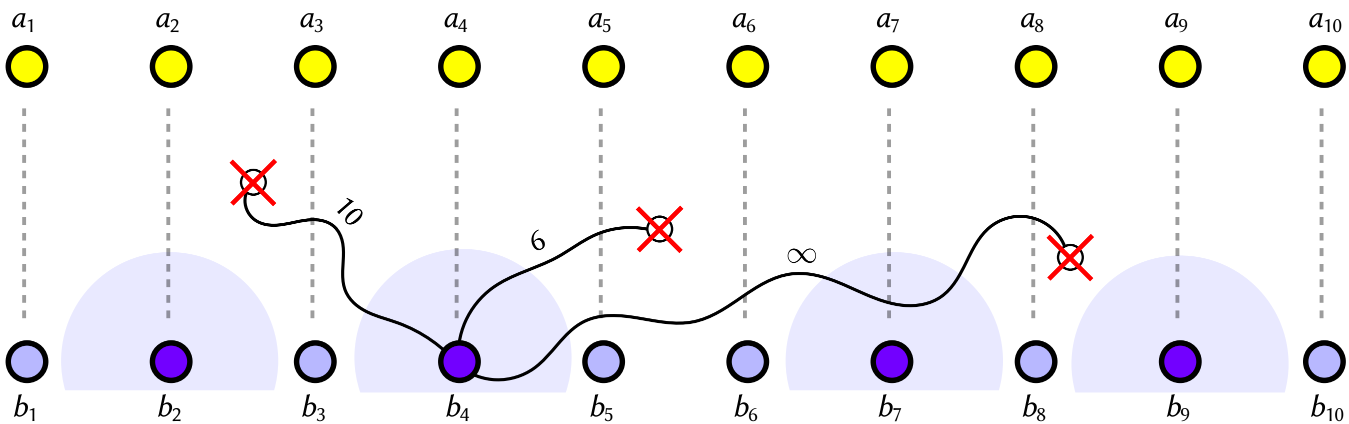
Semi-ladders

From **uqw** we get:

- set S satisfying $|S| \leqslant s$; and
- $B \subseteq \{b_1, \ldots, b_\ell\}$ s.t. $|B| > 2(d+1)^s$ and $\mathrm{dist}_{G-S}(b_i, b_j) > 2d$ for $b_i, b_j \in B$.

For $b_i \in B$, let $\pi_i \colon S \to \{1, \ldots, d, \infty\}$ be its **distance-d profile** on S:

$$\pi_i(v) = \begin{cases} \mathrm{dist}(b_i, v) & \text{if } \leqslant d; \\ \infty & \text{otherwise.} \end{cases}$$



Semi-ladders

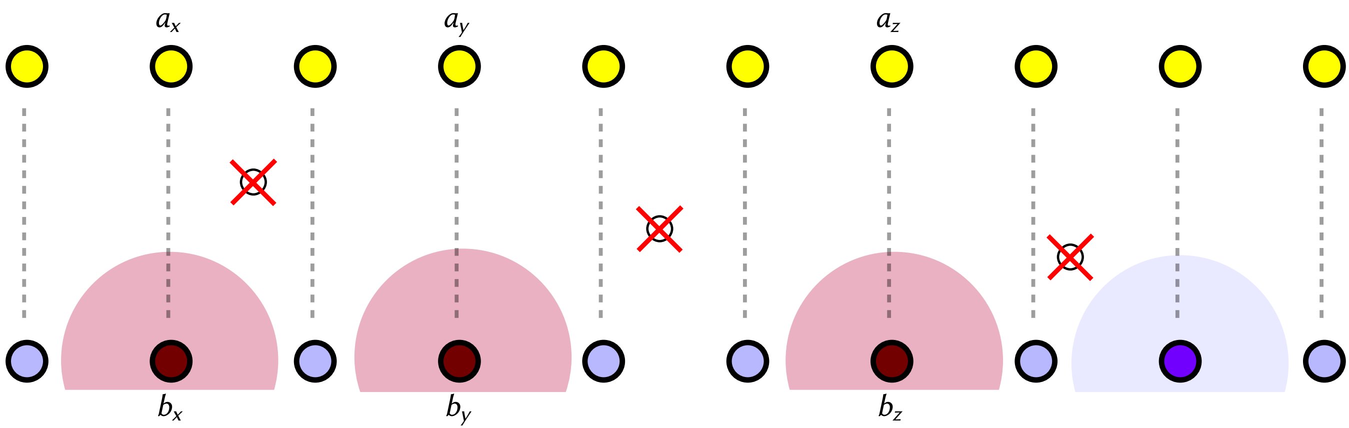
From **uqw** we get:

- set S satisfying $|S| \leqslant s$; and
- $B \subseteq \{b_1, \ldots, b_\ell\}$ s.t. $|B| > 2(d+1)^s$ and $\mathrm{dist}_{G-S}(b_i, b_j) > 2d$ for $b_i, b_j \in B$.

For $b_i \in B$, let $\pi_i \colon S \to \{1, \ldots, d, \infty\}$ be its **distance-d profile** on S:

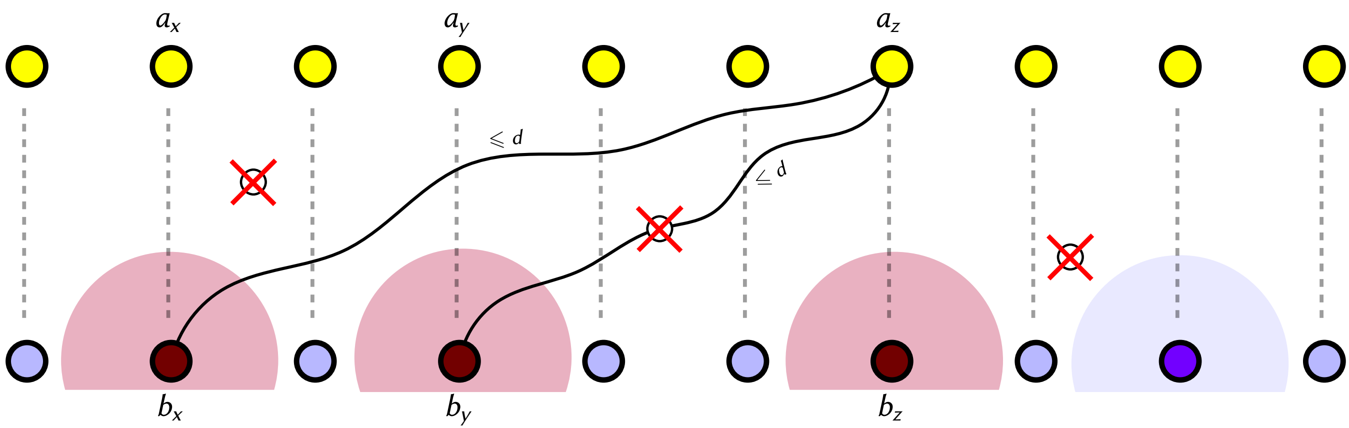
$$\pi_i(v) = \begin{cases} \mathrm{dist}(b_i, v) & \text{if } \leqslant d; \\ \infty & \text{otherwise.} \end{cases}$$

Only $(d+1)^s$ possible profiles $\quad\Rightarrow\quad \exists\ b_x, b_y, b_z$ with same profile.



Semi-ladders

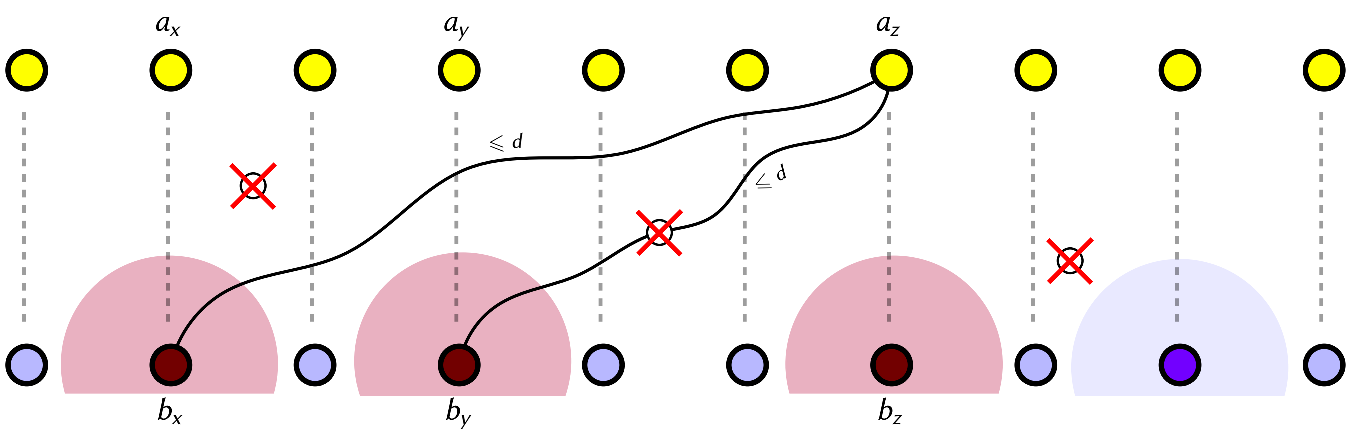
There are a_z-to-b_x and a_z-to-b_y paths of length $\leqslant d$.



Semi-ladders

There are a_z-to-b_x and a_z-to-b_y paths of length $\leqslant d$.

One of them needs to intersect S, say the a_z-to-b_y path.

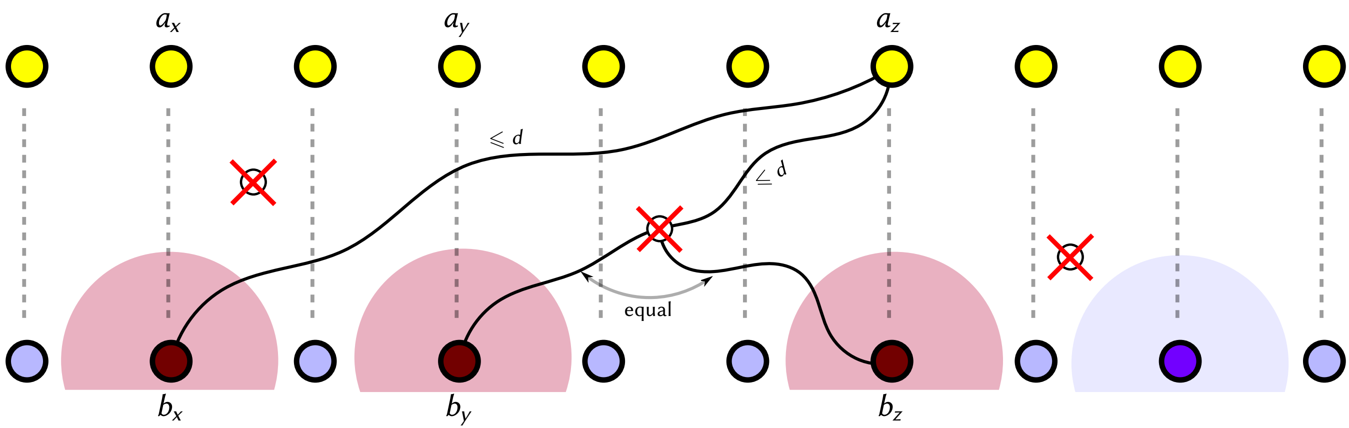


Semi-ladders

There are a_z-to-b_x and a_z-to-b_y paths of length $\leqslant d$.

One of them needs to intersect S, say the a_z-to-b_y path.

$\pi_y = \pi_z \qquad \Rightarrow \qquad$ Same distances to the intersection point.



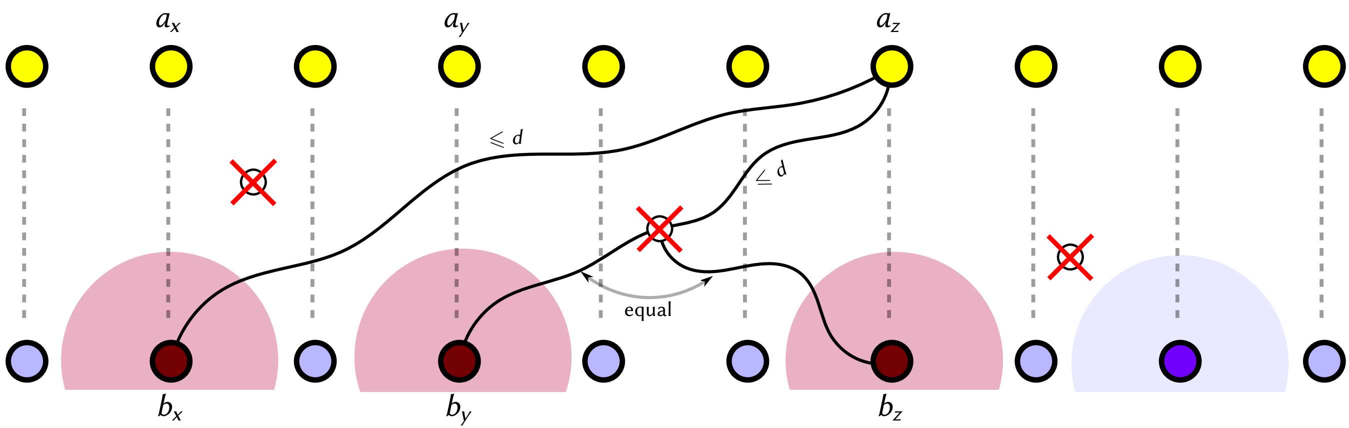
Semi-ladders

There are a_z-to-b_x and a_z-to-b_y paths of length $\leqslant d$.

One of them needs to intersect S, say the a_z-to-b_y path.

$\pi_y = \pi_z \qquad \Rightarrow \qquad$ Same distances to the intersection point.

We conclude that $\mathrm{dist}(a_z, b_z) \leqslant d$. **Contradiction.**



Semi-ladders

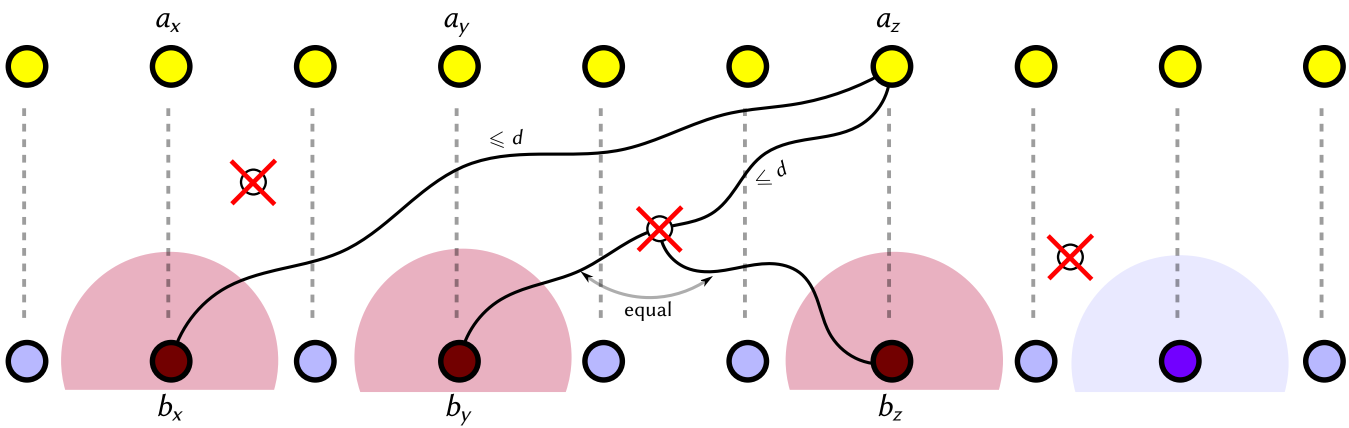
There are a_z-to-b_x and a_z-to-b_y paths of length $\leqslant d$.

One of them needs to intersect S, say the a_z-to-b_y path.

$\pi_y = \pi_z \qquad \Rightarrow \qquad$ Same distances to the intersection point.

We conclude that $\mathrm{dist}(a_z, b_z) \leqslant d$. **Contradiction.**

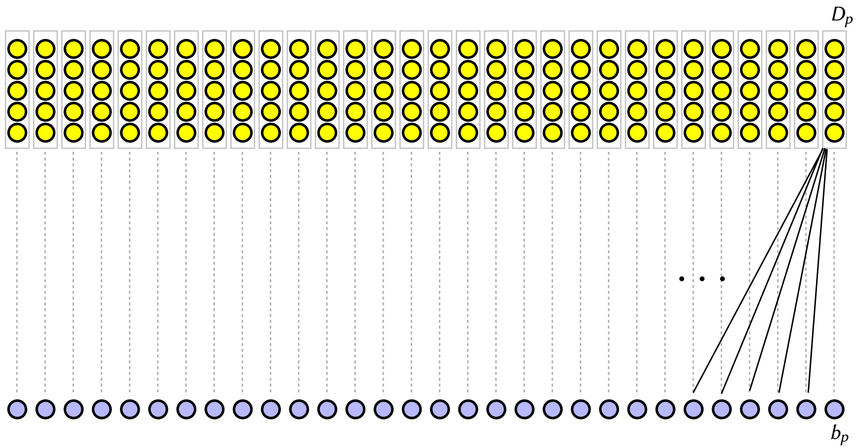
Cor: Maximum semi-ladder order is $\ell := N_{2d}(2(d+1)^{s_{2d}})$.



Case $k > 1$

Claim

For $k > 1$, the number of rounds is $< k^{\ell+1}$, where ℓ is the bound for $k = 1$.

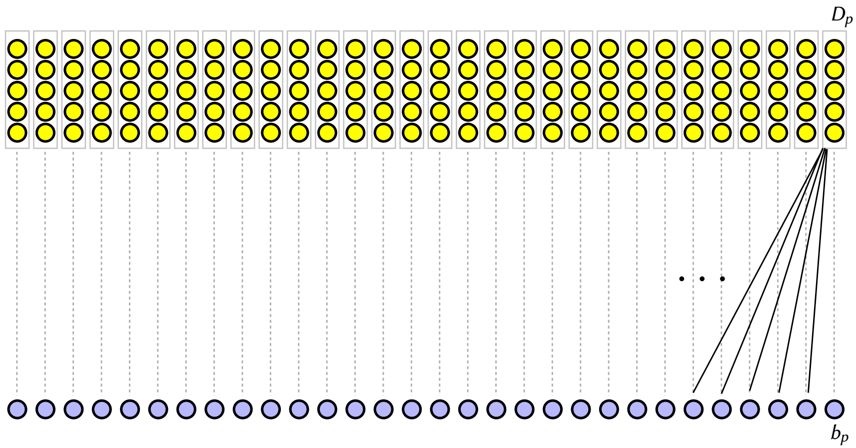


Case $k > 1$

Claim

For $k > 1$, the number of rounds is $< k^{\ell+1}$, where ℓ is the bound for $k = 1$.

Suppose the Algorithm performs $p = k^{\ell+1}$ rounds.



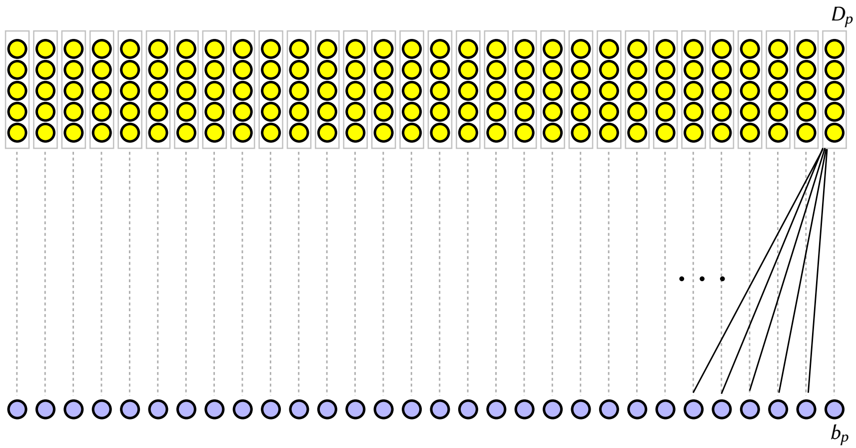
Case $k > 1$

Claim

For $k > 1$, the number of rounds is $< k^{\ell+1}$, where ℓ is the bound for $k = 1$.

Suppose the Algorithm performs $p = k^{\ell+1}$ rounds.

- D_p dist-d dominates $\{b_1, \ldots, b_{p-1}\}$.



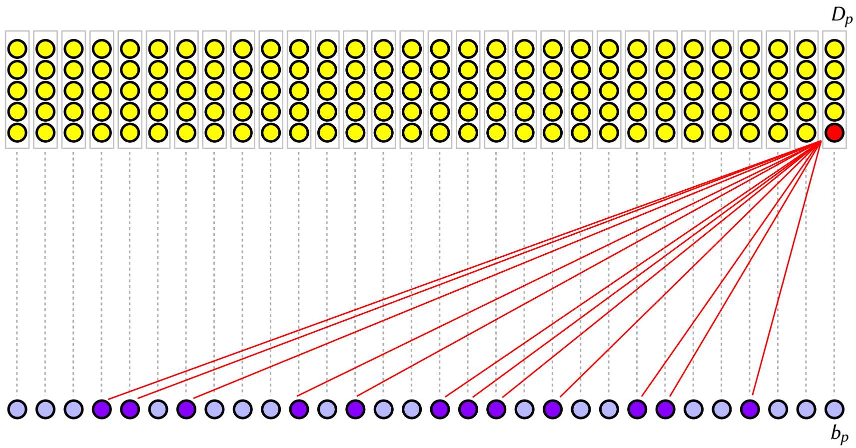
Case $k > 1$

Claim

For $k > 1$, the number of rounds is $< k^{\ell+1}$, where ℓ is the bound for $k = 1$.

Suppose the Algorithm performs $p = k^{\ell+1}$ rounds.

- D_p dist-d dominates $\{b_1, \ldots, b_{p-1}\}$.
- **Hence:** Some $a_p \in D_p$ dist-d dominates $\frac{1}{k}$ fraction of $\{b_1, \ldots, b_{p-1}\}$.



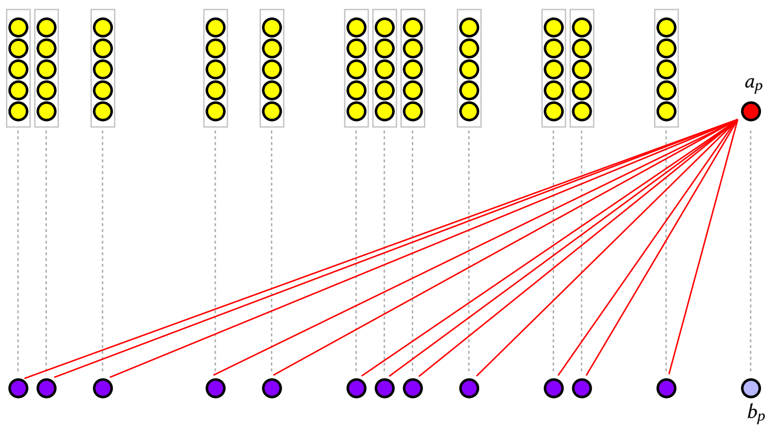
Case $k > 1$

Claim

For $k > 1$, the number of rounds is $< k^{\ell+1}$, where ℓ is the bound for $k = 1$.

Suppose the Algorithm performs $p = k^{\ell+1}$ rounds.

- D_p dist-d dominates $\{b_1, \ldots, b_{p-1}\}$.
- **Hence:** Some $a_p \in D_p$ dist-d dominates $\frac{1}{k}$ fraction of $\{b_1, \ldots, b_{p-1}\}$.
- Restrict attention to those $\geqslant k^\ell$ vertices and **continue**.



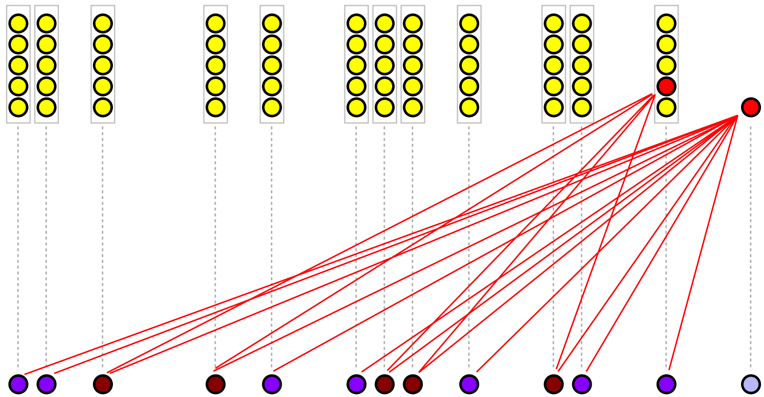
Case $k > 1$

Claim

For $k > 1$, the number of rounds is $< k^{\ell+1}$, where ℓ is the bound for $k = 1$.

Suppose the Algorithm performs $p = k^{\ell+1}$ rounds.

- D_p dist-d dominates $\{b_1, \ldots, b_{p-1}\}$.
- **Hence:** Some $a_p \in D_p$ dist-d dominates $\frac{1}{k}$ fraction of $\{b_1, \ldots, b_{p-1}\}$.
- Restrict attention to those $\geqslant k^\ell$ vertices and **continue**.



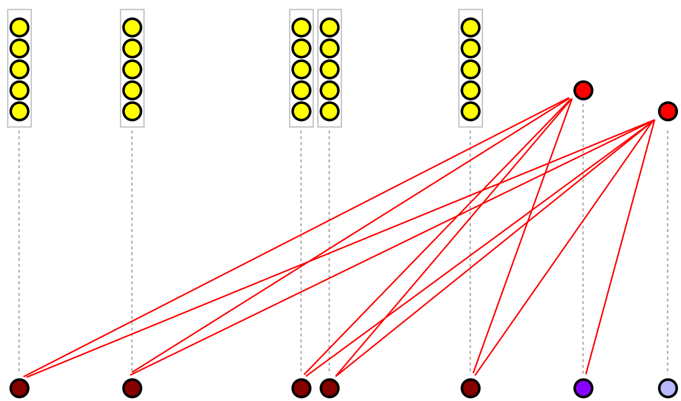
Case $k > 1$

Claim

For $k > 1$, the number of rounds is $< k^{\ell+1}$, where ℓ is the bound for $k = 1$.

Suppose the Algorithm performs $p = k^{\ell+1}$ rounds.

- D_p dist-d dominates $\{b_1, \ldots, b_{p-1}\}$.
- **Hence:** Some $a_p \in D_p$ dist-d dominates $\frac{1}{k}$ fraction of $\{b_1, \ldots, b_{p-1}\}$.
- Restrict attention to those $\geqslant k^{\ell}$ vertices and **continue**.



Case $k > 1$

Claim

For $k > 1$, the number of rounds is $< k^{\ell+1}$, where ℓ is the bound for $k = 1$.

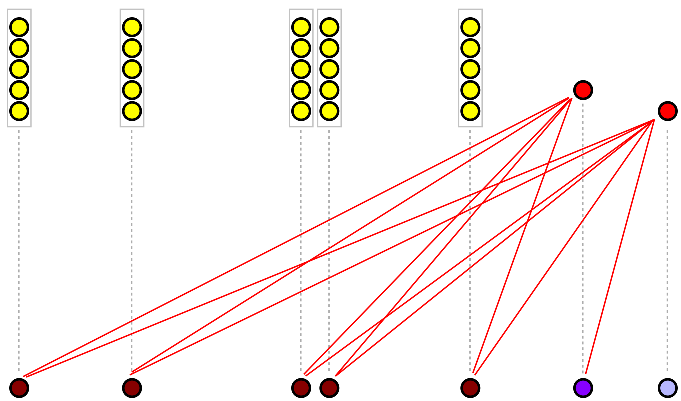
Suppose the Algorithm performs $p = k^{\ell+1}$ rounds.

– D_p dist-d dominates $\{b_1, \ldots, b_{p-1}\}$.

– **Hence:** Some $a_p \in D_p$ dist-d dominates $\frac{1}{k}$ fraction of $\{b_1, \ldots, b_{p-1}\}$.

– Restrict attention to those $\geqslant k^{\ell}$ vertices and **continue**.

$\ell + 1$ rounds $\rightsquigarrow$ a semi-ladder of order $\ell + 1$ **Contradiction.**



Ladders and stability

Ladders and stability

To define **semi-ladders**, we used predicate $\varphi(x, y) =$ "$\text{dist}(x, y) \leqslant d$".

Ladders and stability

To define **semi-ladders**, we used predicate $\varphi(x, y) =$ "$\mathrm{dist}(x, y) \leqslant d$".

Idea: Replace distance checks with any **first-order** predicate.

Ladders and stability

To define **semi-ladders**, we used predicate $\varphi(x, y) =$ "$\mathrm{dist}(x, y) \leqslant d$".

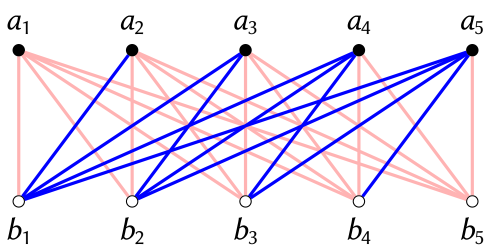
Idea: Replace distance checks with any **first-order** predicate.

Definition

Let G be a graph and $\varphi(x, y)$ be an FO formula.

A **φ-ladder** in G is a pair of sequences $a_1, \ldots, a_\ell$ and $b_1, \ldots, b_\ell$ such that

$$G \models \varphi(a_i, b_j) \qquad \Leftrightarrow \qquad i > j.$$



Ladders and stability

To define **semi-ladders**, we used predicate $\varphi(x, y) =$ “$\mathrm{dist}(x, y) \leqslant d$“.

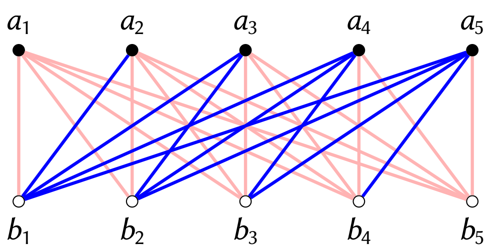
Idea: Replace distance checks with any **first-order** predicate.

Definition

Let G be a graph and $\varphi(x, y)$ be an FO formula.

A **φ-ladder** in G is a pair of sequences $a_1, \ldots, a_\ell$ and $b_1, \ldots, b_\ell$ such that

$$G \models \varphi(a_i, b_j) \qquad \Leftrightarrow \qquad i > j.$$



φ-ladder $\leftrightsquigarrow$ **linear order**

Ladders and stability

To define **semi-ladders**, we used predicate $\varphi(x, y) =$ "$\mathrm{dist}(x, y) \leqslant d$".

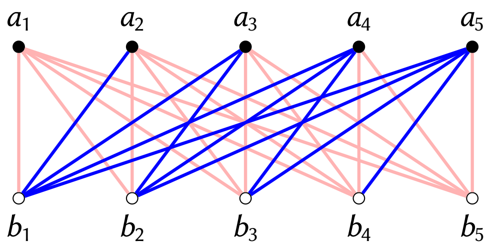
Idea: Replace distance checks with any **first-order** predicate.

Definition

Let G be a graph and $\varphi(x, y)$ be an FO formula.

A **φ-ladder** in G is a pair of sequences $a_1, \ldots, a_\ell$ and $b_1, \ldots, b_\ell$ such that

$$G \models \varphi(a_i, b_j) \qquad \Leftrightarrow \qquad i > j.$$



φ-ladder $\rightsquigarrow$ **linear order**

Definition

A class $\mathcal{C}$ is **stable** if for every FO formula $\varphi(x, y)$, there is a **finite** upper bound on the orders of **φ-ladders** in graphs from $\mathcal{C}$.

Beyond Sparsity

Theorem (Adler & Adler; Podewski & Ziegler)

Every **nowhere dense** class is **stable**.

Every **subgraph-closed stable** class is **nowhere dense**.

Beyond Sparsity

Theorem (Adler & Adler; Podewski & Ziegler)

Every **nowhere dense** class is **stable**.

Every **subgraph-closed** **stable** class is **nowhere dense**.

There are many more **stable** classes than **nowhere dense**:

Beyond Sparsity

Theorem (Adler & Adler; Podewski & Ziegler)

Every **nowhere dense** class is **stable**.

Every **subgraph-closed stable** class is **nowhere dense**.

There are many more **stable** classes than **nowhere dense**:

– For an FO formula $\varphi(x, y)$ and graph G, we define:

$$G^{\varphi} \coloneqq (\ V(G),\ \{uv \colon G \models \varphi(u, v)\}\ ).$$

Beyond Sparsity

Theorem (Adler & Adler; Podewski & Ziegler)

Every **nowhere dense** class is **stable**.

Every **subgraph-closed stable** class is **nowhere dense**.

There are many more **stable** classes than **nowhere dense**:

– For an FO formula $\varphi(x, y)$ and graph G, we define:

$$G^{\varphi} := (\ V(G),\ \{uv\colon G \models \varphi(u, v)\}\ ).$$

– **Ex:** graph powers, complementation,...

Beyond Sparsity

Theorem (Adler & Adler; Podewski & Ziegler)

Every **nowhere dense** class is **stable**.

Every **subgraph-closed** **stable** class is **nowhere dense**.

There are many more **stable** classes than **nowhere dense**:

- For an FO formula $\varphi(x, y)$ and graph G, we define:

$$G^{\varphi} := (\ V(G),\ \{uv\colon\ G \models \varphi(u, v)\}\ ).$$

- **Ex:** graph powers, complementation,...
- For a class $\mathcal{C}$, we define:

$$\mathcal{C}^{\varphi} := \{\ G^{\varphi}\ :\ \ G \in \mathcal{C}\ \}.$$

Beyond Sparsity

Theorem (Adler & Adler; Podewski & Ziegler)

Every **nowhere dense** class is **stable**.

Every **subgraph-closed stable** class is **nowhere dense**.

There are many more **stable** classes than **nowhere dense**:

– For an FO formula $\varphi(x, y)$ and graph G, we define:

$$G^{\varphi} := (\ V(G),\ \{uv \colon G \models \varphi(u, v)\}\ ).$$

– **Ex:** graph powers, complementation,...

– For a class $\mathcal{C}$, we define:

$$\mathcal{C}^{\varphi} := \{\ G^{\varphi}\ :\ \ G \in \mathcal{C}\ \}.$$

– If $\mathcal{C}$ is **nowhere dense**, then $\mathcal{C}^{\varphi}$ is called **structurally nowhere dense**.

Beyond Sparsity

Theorem (Adler & Adler; Podewski & Ziegler)

Every **nowhere dense** class is **stable**.

Every **subgraph-closed stable** class is **nowhere dense**.

There are many more **stable** classes than **nowhere dense**:

– For an FO formula $\varphi(x, y)$ and graph G, we define:

$$G^{\varphi} \coloneqq (\ V(G),\ \{uv \colon G \models \varphi(u, v)\}\ ).$$

– **Ex:** graph powers, complementation,...

– For a class $\mathcal{C}$, we define:

$$\mathcal{C}^{\varphi} \coloneqq \{\ G^{\varphi}\ :\ G \in \mathcal{C}\ \}.$$

– If $\mathcal{C}$ is **nowhere dense**, then $\mathcal{C}^{\varphi}$ is called **structurally nowhere dense**.

Fact. Structurally nowhere dense $\Rightarrow$ **Stable**

Beyond Sparsity

Theorem (Adler & Adler; Podewski & Ziegler)

Every **nowhere dense** class is **stable**.

Every **subgraph-closed stable** class is **nowhere dense**.

There are many more **stable** classes than **nowhere dense**:

– For an FO formula $\varphi(x, y)$ and graph G, we define:

$$G^\varphi := (\ V(G),\ \{uv\colon\ G \models \varphi(u, v)\}\ ).$$

– **Ex:** graph powers, complementation,...

– For a class $\mathcal{C}$, we define:

$$\mathcal{C}^\varphi := \{\ G^\varphi\ :\ \ G \in \mathcal{C}\ \}.$$

– If $\mathcal{C}$ is **nowhere dense**, then $\mathcal{C}^\varphi$ is called **structurally nowhere dense**.

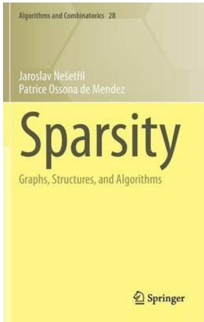
Fact. **Structurally nowhere dense** $\quad\Rightarrow\quad$ **Stable**

Goal: A **theory** of **well-structured dense graphs**.

Further reading

Further reading

Monograph **Sparsity** of Nešetřil and Ossona de Mendez

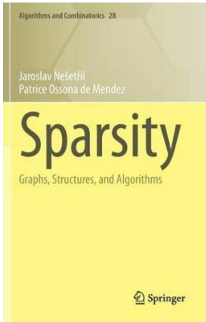


Further reading

Monograph **Sparsity** of Nešetřil and Ossona de Mendez

Lecture notes and **tutorials** at `www.mimuw.edu.pl/~mp248287/sparsity2`

Further reading



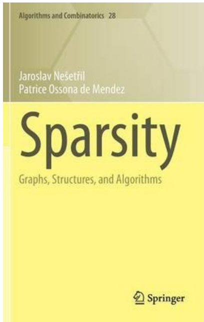
Monograph **Sparsity** of Nešetřil and Ossona de Mendez

Lecture notes and **tutorials** at `www.mimuw.edu.pl/~mp248287/sparsity2`

– Hopefully, one day they will be turned into a book.

Further reading

Monograph **Sparsity** of Nešetřil and Ossona de Mendez

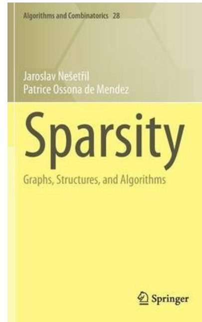


Lecture notes and **tutorials** at `www.mimuw.edu.pl/~mp248287/sparsity2`

– Hopefully, one day they will be turned into a book.

Video recordings of the lectures (link on the website).

Further reading



Monograph **Sparsity** of Nešetřil and Ossona de Mendez

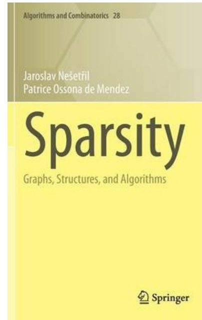
Lecture notes and **tutorials** at `www.mimuw.edu.pl/~mp248287/sparsity2`

– Hopefully, one day they will be turned into a book.

Video recordings of the lectures (link on the website).

Introduction through **exercises** for:

Further reading



Monograph **Sparsity** of Nešetřil and Ossona de Mendez

Lecture notes and **tutorials** at `www.mimuw.edu.pl/~mp248287/sparsity2`

– Hopefully, one day they will be turned into a book.

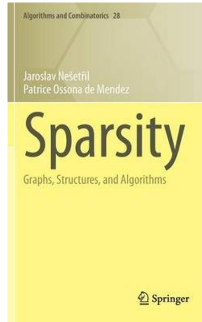
Video recordings of the lectures (link on the website).

Introduction through **exercises** for:

– PhD students of ALGOMANET:

`https://mimuw.edu.pl/~mp248287/sparsity2/algomanet.html`

Further reading



Monograph **Sparsity** of Nešetřil and Ossona de Mendez

Lecture notes and **tutorials** at `www.mimuw.edu.pl/~mp248287/sparsity2`

– Hopefully, one day they will be turned into a book.

Video recordings of the lectures (link on the website).

Introduction through **exercises** for:

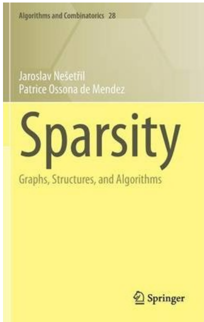
– PhD students of ALGOMANET:

`https://mimuw.edu.pl/~mp248287/sparsity2/algomanet.html`

– high school students at *Math Beyond Limits 2019*:

`https://mimuw.edu.pl/~mp248287/sparsity2/mbl19-sparsity.pdf`

Further reading



Monograph **Sparsity** of Nešetřil and Ossona de Mendez

Lecture notes and **tutorials** at `www.mimuw.edu.pl/~mp248287/sparsity2`

– Hopefully, one day they will be turned into a book.

Video recordings of the lectures (link on the website).

Introduction through **exercises** for:

– PhD students of ALGOMANET:

`https://mimuw.edu.pl/~mp248287/sparsity2/algomanet.html`

– high school students at *Math Beyond Limits 2019*:

`https://mimuw.edu.pl/~mp248287/sparsity2/mbl19-sparsity.pdf`

Thank you for the attention!