

# Invitation to Sparsity

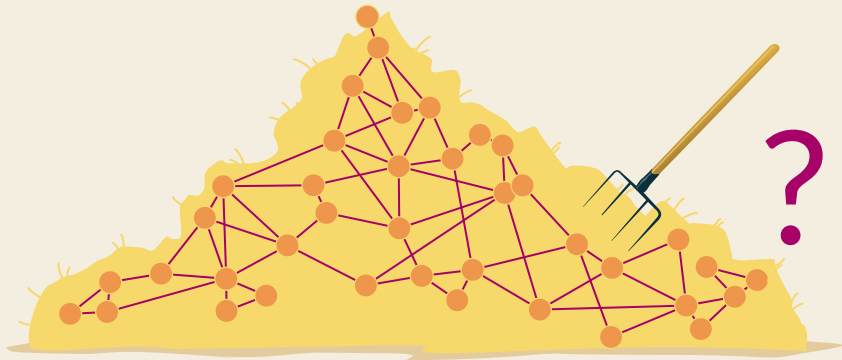
## Algorithmic aspects I

**Felix Reidl**  
Birkbeck College  
[felix.reidl@gmail.com](mailto:felix.reidl@gmail.com)

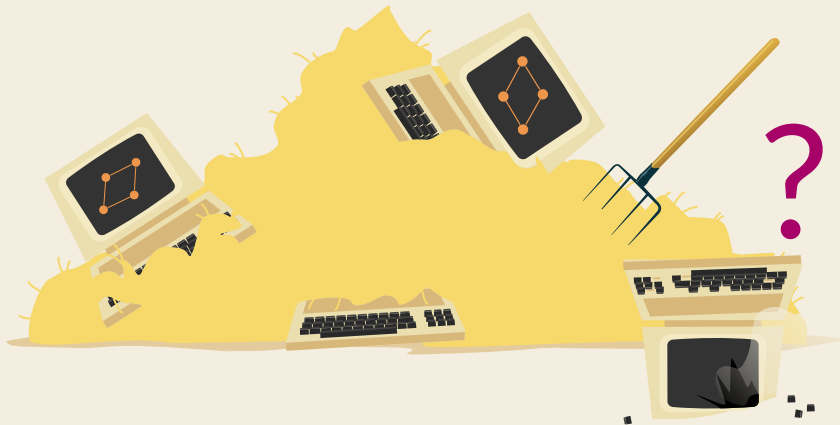
**CanaDAM 2021**  
May 26 2021



How many  are in a

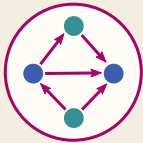


How many  are in a

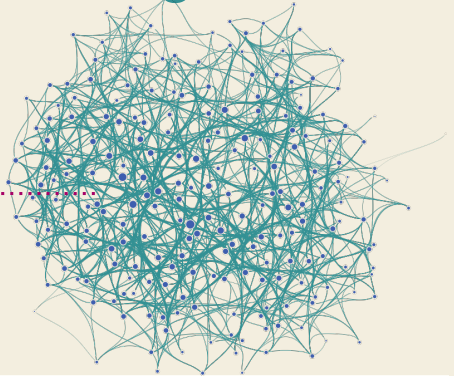


# Motif-counting

We want to count the number of times a given **motif graph**



appears in a larger host graph (network).



Motifs that appear more often **than expected** *might* play an important **function** in the network.

Milo R, Shen-Orr S, Itzkovitz S, Kashtan N, Chklovskii D, Alon U.

**Network motifs: simple building blocks of complex networks.**

Science. 2002 Oct 25;298(5594):824-7.

Ribeiro P, Silva F, Kaiser M. **Strategies for network motifs discovery.**

InE-Science, 2009. e-Science'09. Fifth IEEE International Conference on 2009 Dec 9 (pp. 80-87). IEEE.

# Counting subgraphs\*

(INDUCED) SUBGRAPH COUNTING

**Input:** A host graph  $G$  and a pattern graph  $H$ .

**Problem:** How often does  $H$  appear in  $G$  as a (induced) subgraph?

# Counting subgraphs\*

## (INDUCED) SUBGRAPH COUNTING

**Input:** A host graph  $G$  and a pattern graph  $H$ .

**Problem:** How often does  $H$  appear in  $G$  as a (induced) subgraph?

Bad news

#W[1]-hard even if the pattern is a clique

Flum J, Grohe M. **The parameterized complexity of counting problems.**  
SIAM Journal on Computing. 2004;33(4):892-922.

#W[1]-hard even if the pattern is a path

Chen Y, Flum J. **On parameterized path and chordless path problems.**  
In Twenty-Second Annual IEEE Conference on Computational Complexity (CCC'07) 2007 Jun 13 (pp. 250-263). IEEE.

# Counting subgraphs\*

## (INDUCED) SUBGRAPH COUNTING

**Input:** A host graph  $G$  and a pattern graph  $H$ .

**Problem:** How often does  $H$  appear in  $G$  as a (induced) subgraph?

Bad news

#W[1]-hard even if the pattern is a clique

Flum J, Grohe M. **The parameterized complexity of counting problems.** SIAM Journal on Computing. 2004;33(4):892-922.

#W[1]-hard even if the pattern is a path

Chen Y, Flum J. **On parameterized path and chordless path problems.** In Twenty-Second Annual IEEE Conference on Computational Complexity (CCC'07) 2007 Jun 13 (pp. 250-263). IEEE.

No  $f(|H|)$  poly( $|G|$ ) algorithm unless other weird things are true.

Flum J, Grohe M. **Parameterized complexity theory.** Springer Science & Business Media; 2006 May 1.

# Counting subgraphs\*

## (INDUCED) SUBGRAPH COUNTING

**Input:** A host graph  $G$  and a pattern graph  $H$ .

**Problem:** How often does  $H$  appear in  $G$  as a (induced) subgraph?

Worse news

Even *finding* a clique of size  $k$  is probably not possible in time  $f(k)|G|^{o(k)}$ .

Chen J, Huang X, Kanj IA, Xia G. **Strong computational lower bounds via parameterized complexity.** Journal of Computer and System Sciences. 2006 Dec 1;72(8):1346-67.



# Counting subgraphs\*

## (INDUCED) SUBGRAPH COUNTING

**Input:** A host graph  $G$  and a pattern graph  $H$ .

**Problem:** How often does  $H$  appear in  $G$  as a (induced) subgraph?

Worse news

Even *finding* a clique of size  $k$  is probably not possible in time  $f(k)|G|^{o(k)}$ .

Chen J, Huang X, Kanj IA, Xia G. **Strong computational lower bounds via parameterized complexity.** Journal of Computer and System Sciences. 2006 Dec 1;72(8):1346-67.

Unless we make assumptions about the host graph, we cannot do much better than brute-force.

Larger classes



Less

Structure

More

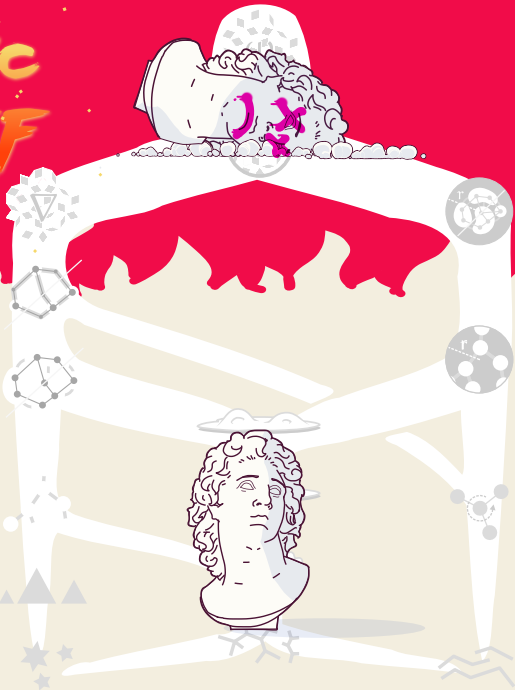


Algorithmic tractability



# ICONOCLASTIC HOT STUFF

— ∞ ∞ —  
The *Classic*  
works on  
sparse graphs  
— ∞ ∞ —



# Parameterised graph invariants

A **graph invariant** is an isomorphism invariant function that maps graphs to  $\mathbb{R}^+$

e.g. density, average degree, clique number, degeneracy treewidth, etc.

A **parameterised graph invariant** is a family of graph invariant  $(f_r)_{r \in \mathbb{N}_0}$ .

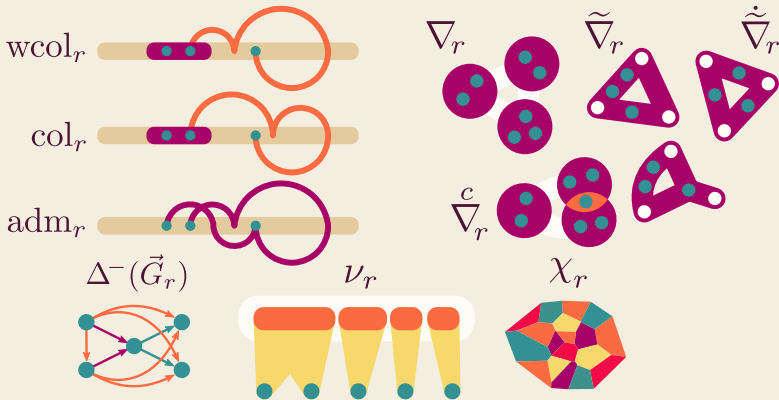
A graph class  $\mathcal{G}$  is  **$f_r$ -bounded** if there exists  $g$  s.t.

$$f_r(\mathcal{G}) = \limsup_{G \in \mathcal{G}} f_r(G) \leq g(r) \text{ for all } r.$$

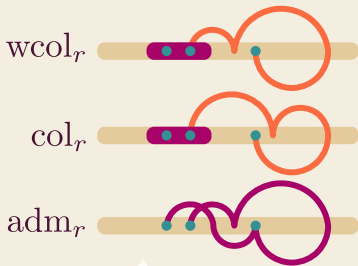
# Bounded expansion

Nešetřil & Ossona de Mendez:

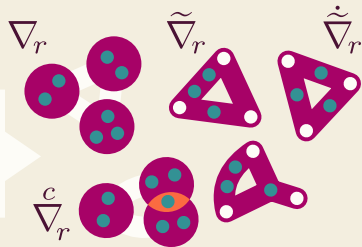
Many notions of  $f_r$ -boundedness are equivalent!



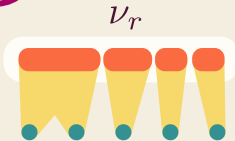
# Bounded expansion



Density of shallow minors



Size of  $r$ -reachable sets in ordering



Normalized number of traces  $r$ -neighbourhoods leave in any subset

$$\Delta^-(\vec{G}_r)$$



In-degree of  $r$ -step (d)tf-augmentation

Number of colours in  $r$ -treedepth colouring

$$\chi_r$$



# The hammer scale

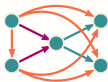


$wcol_r$



$\nu_r$

$\Delta^-(\vec{G}_r)$



$\chi_r$



FO model  
checking



$r$ -Dominating  
Set approx.



splitter games

Usability  
in *practice*

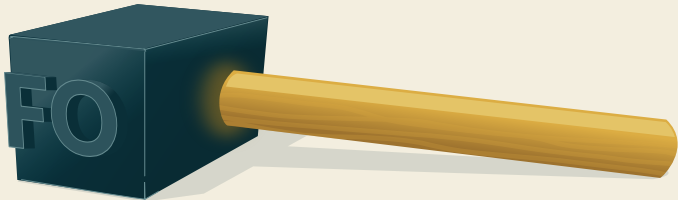


Usability  
in *theory*



# *Part I*

## The **big** hammer





# FO checking/counting/enumeration

**Theorem.** Given a graph  $G$  from a BE class and a FO-sentence  $\phi$ , one can **decide** whether  $G \models \phi$  in time  $O(f(|\phi|) |G|)$ .

Dvořák Z, Král D, Thomas R. **Deciding first-order properties for sparse graphs.** In 2010 IEEE 51st Annual Symposium on Foundations of Computer Science 2010 Oct 23 (pp. 133-142). IEEE.

**Theorem.** Given a graph  $G$  from a BE class and a FO-formula  $\phi(\bar{x})$ , one can **count** the number of tuples  $\bar{a}$  with  $G \models \phi(\bar{a})$  in time  $O(f(|\phi|) |G|)$ .

**Theorem.** Given a graph  $G$  from a BE class and a FO-formula  $\phi(\bar{x})$ , one can compute a data structure in time  $O(f(|\phi|) |G|)$  which **enumerates** all tuples  $\bar{a}$  satisfying  $G \models \phi(\bar{a})$  with constant delay.

Kazana W, Segoufin L. **Enumeration of first-order queries on classes of structures with bounded expansion.** In Proceedings of the 32nd ACM SIGMOD-SIGACT-SIGAI symposium on Principles of database systems 2013 Jun 22 (pp. 297-308).

# FO checking/counting/enumeration

**Theorem.** Given a graph  $G$  from a BE class and a FO-sentence  $\phi$ , one can **decide** whether  $G \models \phi$  in time  $O(f(|\phi|) |G|)$ .

Similar results exist for **nowhere-dense** classes

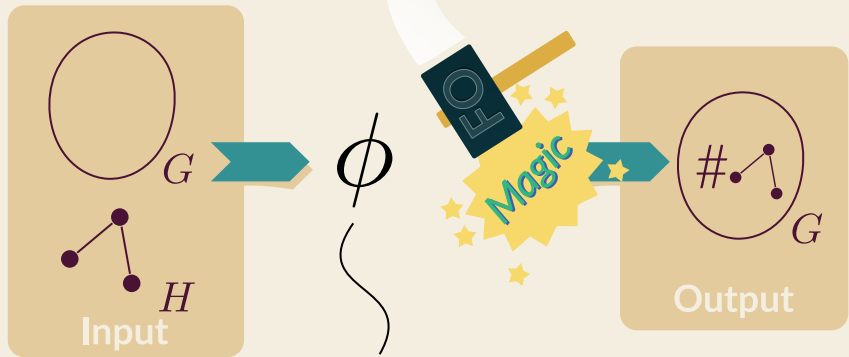
Grohe M, Kreutzer S, Siebertz S. **Deciding first-order properties of nowhere dense graphs.** Journal of the ACM (JACM). 2017 Jun 16;64(3):1-32.

Schweikardt N, Segoufin L, Vigny A. **Enumeration for FO queries over nowhere dense graphs.** In Proceedings of the 37th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems 2018 May 27 (pp. 151-163).

**Theorem.** Given a graph  $G$  from a BE class and a FO-formula  $\phi(\bar{x})$ , one can compute a data structure in time  $O(f(|\phi|) |G|)$  which **enumerates** all tuples  $\bar{a}$  satisfying  $G \models \phi(\bar{a})$  with constant delay.

Kazana W, Segoufin L. **Enumeration of first-order queries on classes of structures with bounded expansion.** In Proceedings of the 32nd ACM SIGMOD-SIGACT-SIGAI symposium on Principles of database systems 2013 Jun 22 (pp. 297-308).

# Counting subgraphs in linear time



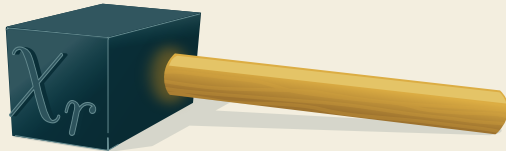
$$\phi(x_1, \dots, x_3) = \underbrace{x_1 E x_2 \wedge x_2 E x_3 \wedge \neg x_1 E x_3}_{O(\|H\|)}$$



**Big hammers don't implement**

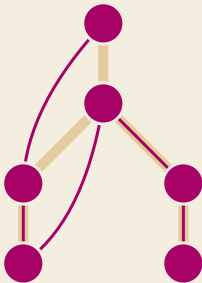
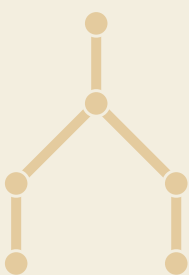
## *Part II*

# The medium hammer



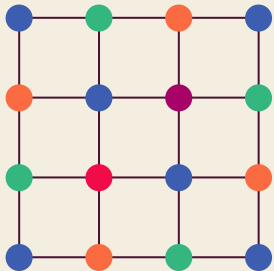
# Treedepth

**Def.** A graph has *treedepth*  $d$  if it is the subgraph of the *closure* of a tree of height  $d$ .



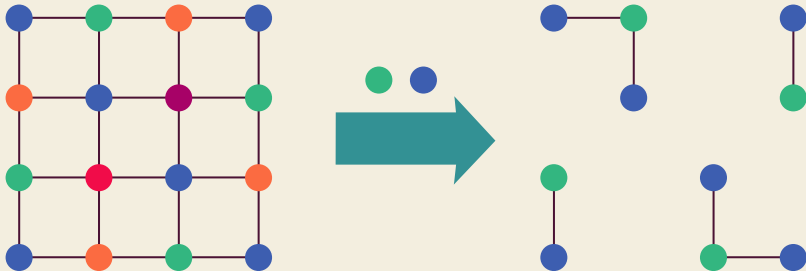
# Low treedepth colourings

A vertex colouring is an **r-treedepth colouring** if every set of  $i < r$  colours induce a subgraph of treedepth  $i$ .



# Low treedepth colourings

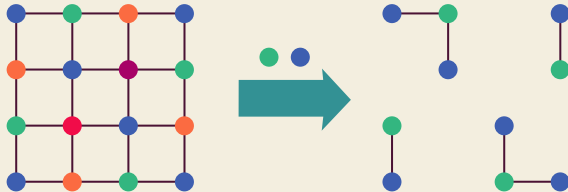
A vertex colouring is an **r-treedepth colouring** if every set of  $i < r$  colours induce a subgraph of treedepth  $i$ .





# Low treedepth colourings

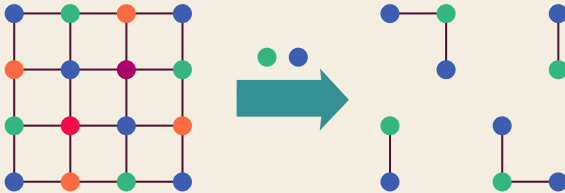
A vertex colouring is an **r-treedepth colouring** if every set of  $i < r$  colours induce a subgraph of treedepth  $i$ .



Define  $\chi_r$  to be the number of colours needed for an r-treedepth colouring.

# Low treedepth colourings

A vertex colouring is an **r-treedepth colouring** if every set of  $i < r$  colours induce a subgraph of treedepth  $i$ .

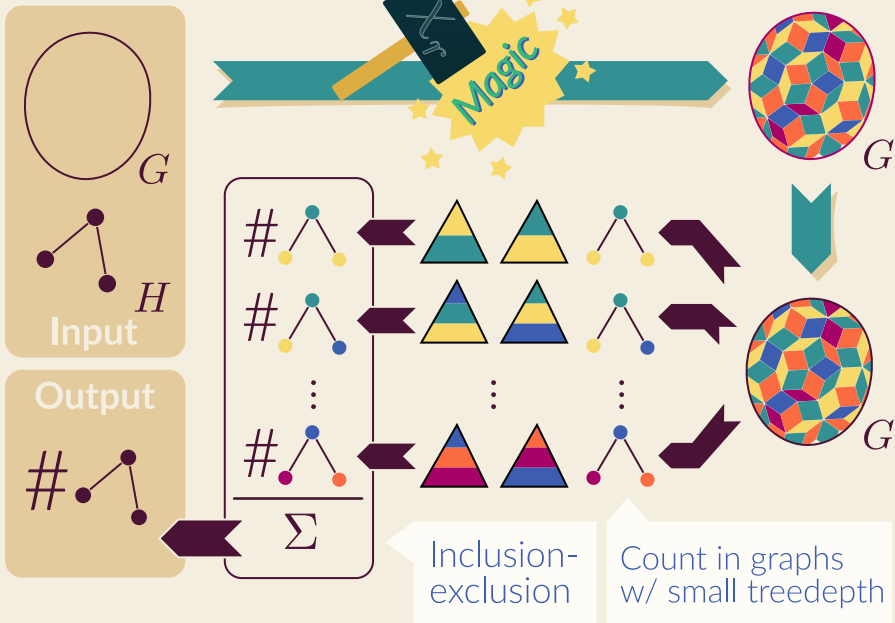


Define  $\chi_r$  to be the number of colours needed for an r-treedepth colouring.

A graph class has bounded expansion iff it is  $\chi_r$ -bounded.



# Subgraph counting using $\chi_r$





# Medium hammers don't scale

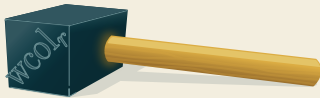
O'Brien MP, Sullivan BD.

Experimental evaluation of counting subgraph isomorphisms  
in classes of bounded expansion.

arXiv preprint arXiv:1712.06690. 2017 Dec 18.

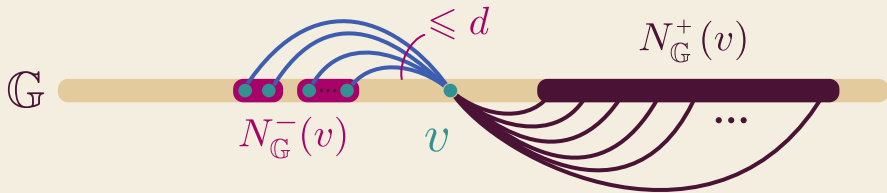
## *Part III*

# The **small**\* hammer



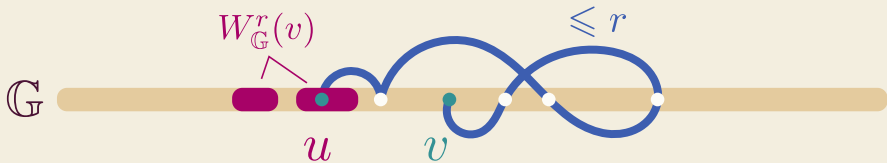
\*small-ish

# Degeneracy



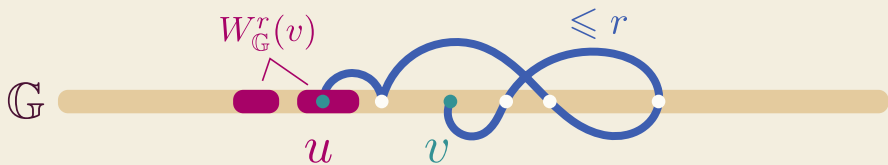
A graph  $G$  is  $d$ -degenerate if there exists a linear ordering  $\mathbb{G}$  of  $G$  such that every vertex has at most  $d$  neighbours to its left.

# Weak colouring & bounded expansion



$u$  is **weakly  $r$ -reachable** from  $v$  if there exists a path from  $v$  to  $u$  of length at most  $r$  such that  $u$  is the path's leftmost vertex.

# Weak colouring & bounded expansion

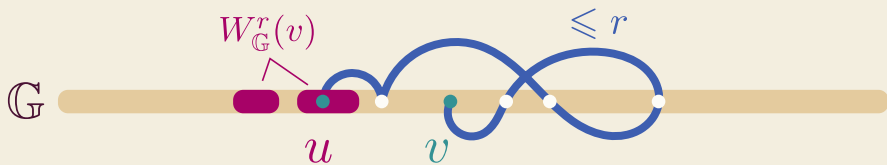


$u$  is **weakly  $r$ -reachable** from  $v$  if there exists a path from  $v$  to  $u$  of length at most  $r$  such that  $u$  is the path's leftmost vertex.

$$\text{wcol}_r(G) := \min_{G \in \Pi(G)} \max_{v \in G} |W_G^r(v)|$$



# Weak colouring & bounded expansion



$u$  is **weakly  $r$ -reachable** from  $v$  if there exists a path from  $v$  to  $u$  of length at most  $r$  such that  $u$  is the path's leftmost vertex.

$$\text{wcol}_r(G) := \min_{G \in \Pi(G)} \max_{v \in G} |W_G^r(v)|$$

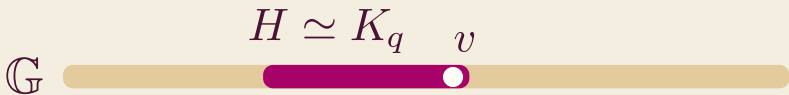


A graph class has bounded expansion iff it is **wcol $_r$ -bounded**.

# Let's start with something easy!

We count cliques in a  $d$ -degenerate graph.

**Observation:** every clique is contained in the left-neighbourhood of its *last* vertex.

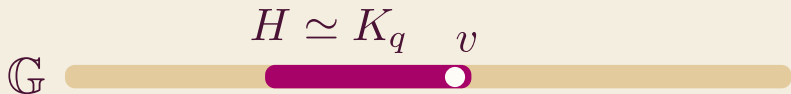


$$V(H) \subseteq N_G^-(v)$$

# Let's start with something easy!

We count cliques in a  $d$ -degenerate graph.

**Observation:** every clique is contained in the left-neighbourhood of its *last* vertex.



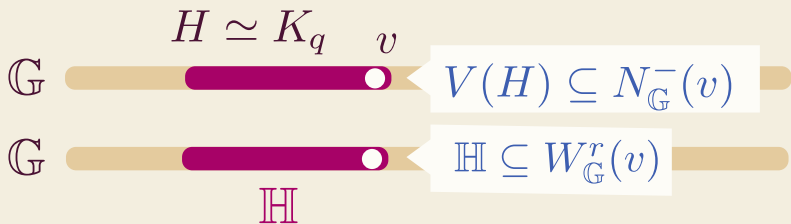
$$V(H) \subseteq N_G^-(v)$$

Therefore we can enumerate all cliques by enumerating all cliques in  $N^-(v)$  for all  $v \in G$ !

$$O(2^d n) \text{ time!}$$

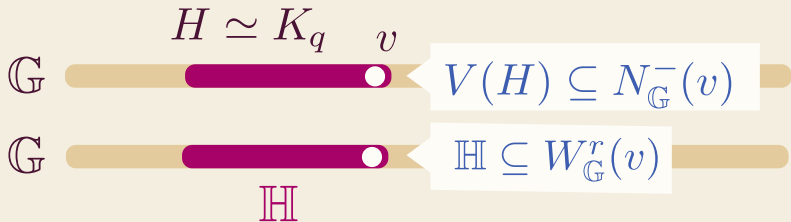
# Does it blend?

Can we 'lift' this algorithm to wcol?



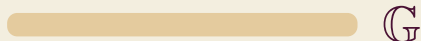
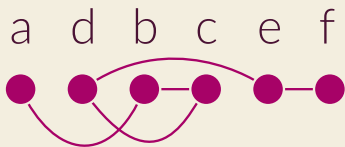
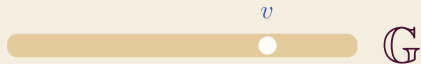
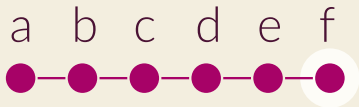
# Does it blend?

Can we 'lift' this algorithm to wcol?

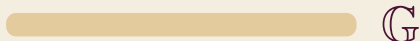
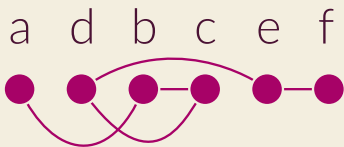
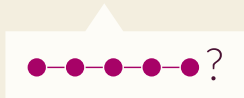
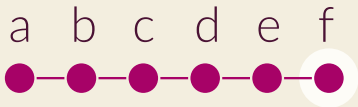


- 1 What is the 'last' vertex of  $H$ ?  
Enumerate all orderings  $\mathbb{H}$  of  $H$ .
- 2 Does  $\mathbb{H} \subseteq W_{\mathbb{G}}^r(v)$  actually hold?  
Only sometimes!

# Two ways to order a $P_6$



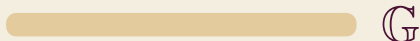
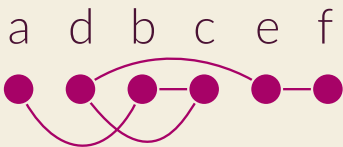
# Two ways to order a $P_6$



# Two ways to order a $P_6$



$$W_{P_6}^5(f)!$$

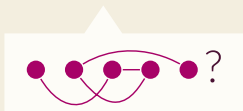
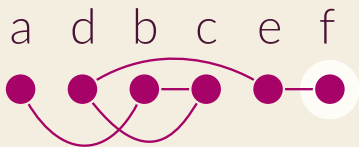




# Two ways to order a $P_6$



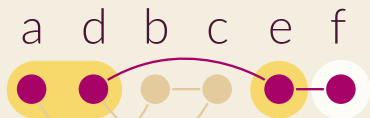
$$W_{P_6}^5(f)!$$



# Two ways to order a $P_6$



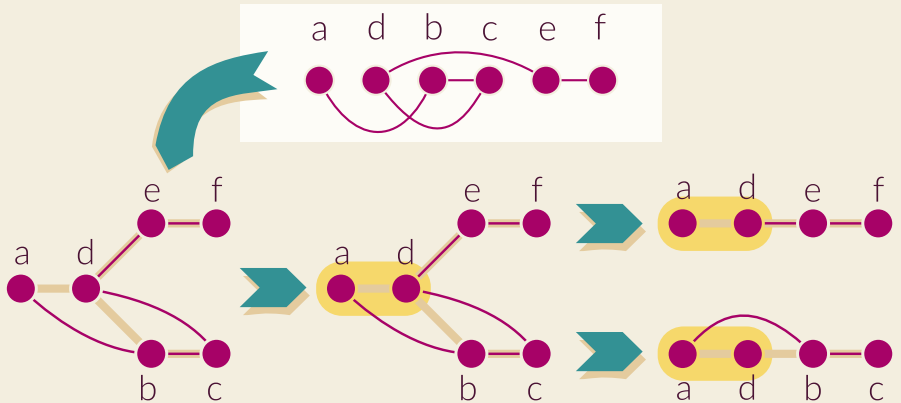
$$W_{P_6}^5(f)!$$



$$W_{P'_6}^5(f) \dots$$



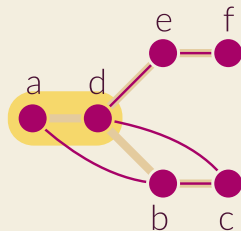
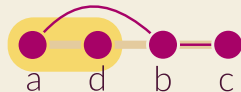
# Decomposition!



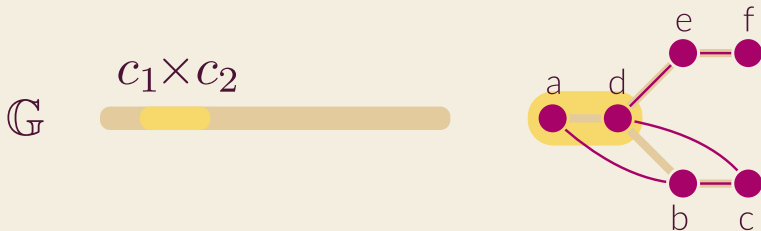
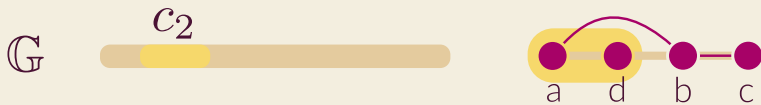
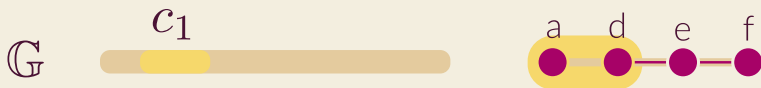
We can count linear pieces!

Progress! These pieces are linear!

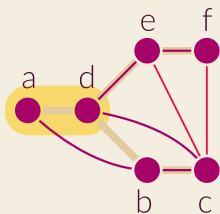
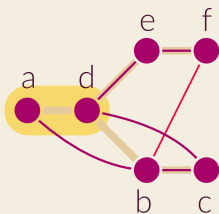
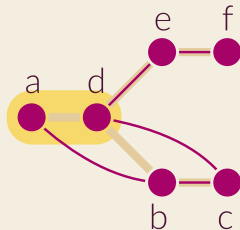
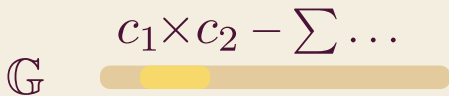
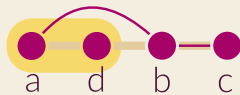
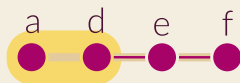
# Count & combine!



# Count & combine!

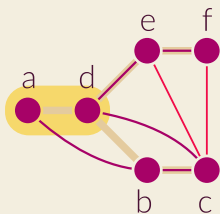
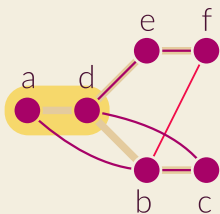
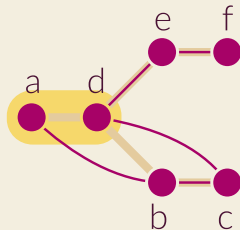
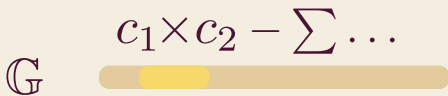
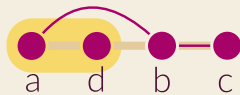
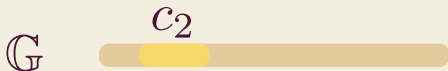


# Count & combine!



...

# Count & combine!

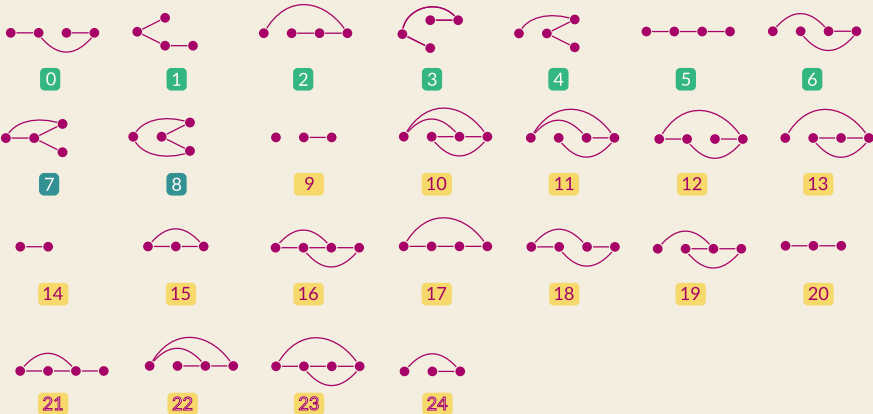


How do we count these graphs?

# Counting $P_4$ s using $wcol_3$

**Lemma 6.** Let  $\mathbf{H} \in \mathcal{H}$  be a (non-linear) pattern relaxation and let  $\mathbf{H}_1 \oplus_{\bar{x}} \mathbf{H}_2 = \mathbf{H}$ . Fix an ordered vertex set  $\bar{y} \in \mathbb{G}$  such that  $\mathbf{H}[\bar{x}] \simeq \mathbb{G}[\bar{y}]$ . Then

$$\#_{\bar{x} \mapsto \bar{y}}(\mathbf{H}, \mathbb{G}) = \#_{\bar{x} \mapsto \bar{y}}(\mathbf{H}_1, \mathbb{G}) \#_{\bar{x} \mapsto \bar{y}}(\mathbf{H}_2, \mathbb{G}) - \sum_{\mathbf{D} \in \mathcal{D}(\mathbf{H}_1, \mathbf{H}_2)} \frac{\#_{\bar{x} \mapsto \bar{x}}(\mathbf{H}, \mathbf{D} \mid \mathbf{H}_1, \mathbf{H}_2) \#_{\bar{x} \mapsto \bar{y}}(\mathbf{D}, \mathbb{G})}{\#_{\bar{x} \mapsto \bar{x}}(\mathbf{D}, \mathbf{D})}.$$



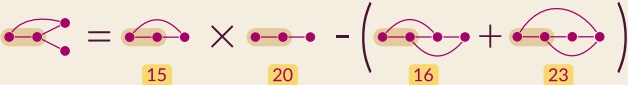



# Counting $P_4$ s using $wcol_3$

**Lemma 6.** Let  $\mathbf{H} \in \mathcal{H}$  be a (non-linear) pattern relaxation and let  $\mathbf{H}_1 \oplus_{\bar{x}} \mathbf{H}_2 = \mathbf{H}$ . Fix an ordered vertex set  $\bar{y} \in \mathbb{G}$  such that  $\mathbf{H}[\bar{x}] \simeq \mathbb{G}[\bar{y}]$ . Then

$$\#_{\bar{x} \mapsto \bar{y}}(\mathbf{H}, \mathbb{G}) = \#_{\bar{x} \mapsto \bar{y}}(\mathbf{H}_1, \mathbb{G}) \#_{\bar{x} \mapsto \bar{y}}(\mathbf{H}_2, \mathbb{G}) - \sum_{\mathbf{D} \in \mathcal{D}(\mathbf{H}_1, \mathbf{H}_2)} \frac{\#_{\bar{x} \mapsto \bar{x}}(\mathbf{H}, \mathbf{D} \mid \mathbf{H}_1, \mathbf{H}_2) \#_{\bar{x} \mapsto \bar{y}}(\mathbf{D}, \mathbb{G})}{\#_{\bar{x} \mapsto \bar{x}}(\mathbf{D}, \mathbf{D})}.$$

4 

7 

8 

Not shown:







## Small hammers might just work!

Nadara W, Pilipczuk M, Rabinovich R, Reidl F, Siebertz S.  
**Empirical evaluation of approximation algorithms for generalized graph coloring and uniform quasi-wideness.**  
Journal of Experimental Algorithmics (JEA). 2019 Dec 10;24:1-34.

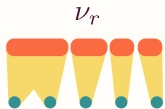
Brown CT, Moritz D, O'Brien MP, Reidl F, Reiter T, Sullivan BD.  
**Exploring neighborhoods in large metagenome assembly graphs using spacegraphcats reveals hidden sequence diversity.** Genome biology. 2020 Dec;21(1):1-6.

[github.com/  
spacegraphcats/  
spacegraphcats](https://github.com/spacegraphcats/spacegraphcats)

Reidl F, Sullivan BD. **A color-avoiding approach to subgraph counting in bounded expansion classes.**  
arXiv preprint arXiv:2001.05236. 2020 Jan 15.

[github.com/  
theoryinpractice/  
mandoline](https://github.com/theoryinpractice/mandoline)

# The hammer scale



$$\Delta^-(\vec{G}_r)$$



FO model  
checking



r-Dominating  
Set approx.



splitter games

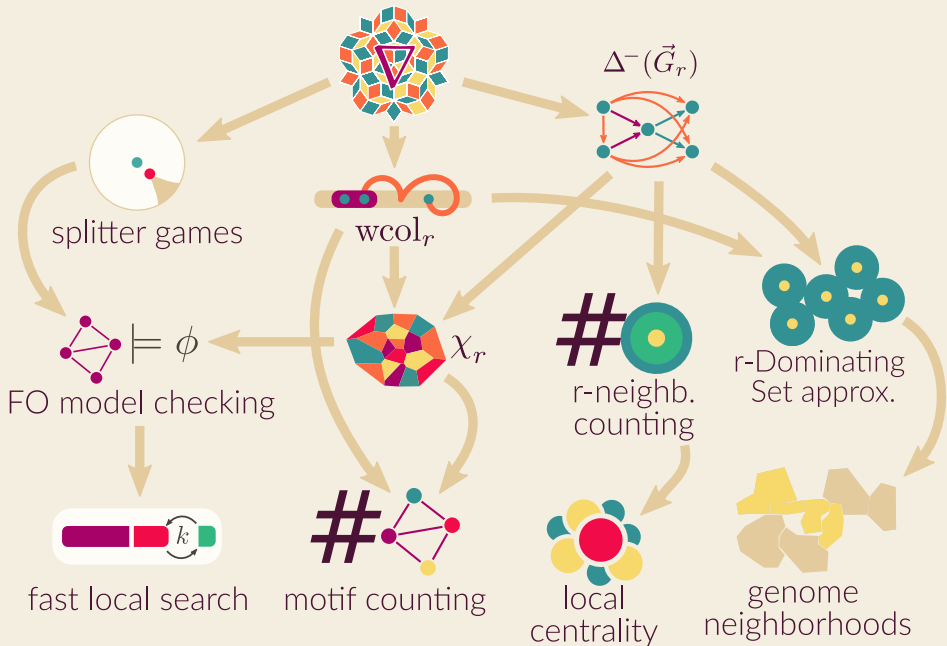
Usability  
in *practice*



Usability  
in *theory*



# Applications & Algorithms



# THANKS!

## Questions?

Next up:

Michał tells you how to  
use sparsity *without*  
*using sparsity!*

Don't miss it!

