Unbounded clique-width in hereditary graph classes

Robert Brignall

Based on joint work with Dan Cocks

Birmingham-Open-Warwick Combinatorics Meeting, OU, 17th December 2024

The Open University

Grid theorems

Grid minor theorem (Robertson & Seymour, 1986)

A minor-closed class of graphs has bounded tree-width if and only if it excludes a planar graph.

Graph minor: delete vertices or edges, and contract edges.
Tree-width: measures how much a graph is like a tree.

Grid theorems

Grid minor theorem (Robertson & Seymour, 1986)

A minor-closed class of graphs has bounded tree-width if and only if it excludes a planar graph.

Graph minor: delete vertices or edges, and contract edges.
Tree-width: measures how much a graph is like a tree.

Grid theorem for vertex minors (Geelen, Kwon, Mccarty, Wollan, 2023)

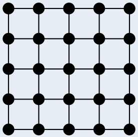
A vertex-minor-closed class of graphs has bounded rank-width if and only if it excludes a circle graph.

Vertex-minor: delete vertices and take 'local complements'.
Rank-width: a graph measure involving ranks of matrices in certain decompositions of a graph.

Grid theorems – alternative statements

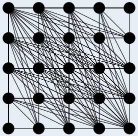
Grid minor theorem (Robertson & Seymour, 1986)

Graphs of large tree-width contain a large grid as a minor.



Grid theorem for vertex minors (Geelen, Kwon, Mccarty, Wollan, 2023)

Graphs of large rank-width contain a large comparability grid as a vertex-minor.



Metatheorems

Theorem (Courcelle, 1990)

Any problem expressible in MSO_2 logic can be solved in linear time on every class of graphs with bounded tree-width.

MSO_2 logic covers problems like existence of perfect matchings, or Hamiltonian cycles.

Metatheorems

Theorem (Courcelle, 1990)

Any problem expressible in MSO_2 logic can be solved in linear time on every class of graphs with bounded tree-width.

MSO_2 logic covers problems like existence of perfect matchings, or Hamiltonian cycles.

Theorem (Courcelle, Makowsky, Rotics, 2000)

Any problem expressible in MSO_1 logic can be solved in linear time on every class of graphs with bounded rank-width.

MSO_1 logic: Weaker than MSO_2, but includes finding a maximum independent set, and deciding k-colourability.

In simple terms

If a collection of graphs has…

…some planar graph as a forbidden minor, lots of graph problems are easy to solve.

In simple terms

If a collection of graphs has…

…some planar graph as a forbidden minor, lots of graph problems are easy to solve.

…some circle graph as a forbidden vertex-minor, not-so-many-lots of graph problems are easy to solve.

In simple terms

If a collection of graphs has...

...some planar graph as a forbidden minor, lots of graph problems are easy to solve.

...some circle graph as a forbidden vertex-minor, not-so-many-lots of graph problems are easy to solve.

Why use anything other than treewidth?

$$\mathrm{tw}(K_n) = n - 1.$$

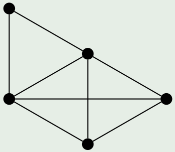
Classes with bounded tree-width can't contain dense graphs.

Is there a ‘grid theorem’ for bounding clique-width in hereditary classes?

Induced subgraphs

- Graph $G = (V, E)$, undirected, simple (no loops, or multiple edges).
- Induced subgraph: $H \leqslant_{\text{ind}} G$ if we can delete vertices (and incident edges) from G to form a graph isomorphic to H.

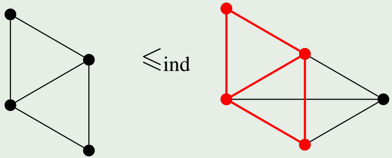
Example (Graphs and induced subgraphs)



Induced subgraphs

- Graph $G = (V, E)$, undirected, simple (no loops, or multiple edges).
- Induced subgraph: $H \leqslant_{\text{ind}} G$ if we can delete vertices (and incident edges) from G to form a graph isomorphic to H.

Example (Graphs and induced subgraphs)



Hereditary classes

Set $\mathcal{C}$ of graphs is hereditary if

$$G \in \mathcal{C} \text{ and } H \leqslant_{\text{ind}} G \text{ implies } H \in \mathcal{C}.$$

'class'

'Closed under induced subgraphs'.

Examples

Forests

Bipartite graphs

Planar graphs

Circle graphs

Permutation graphs

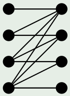
Build-a-graph

You have 4 operations to build a graph:

1. Create a new vertex with a label i.
2. Disjoint union of two previously-constructed graphs.
3. Join all vertices labelled i to all labelled j ($i \neq j$).
4. Relabel every vertex labelled i with j.

Example

Target:

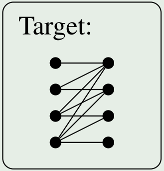


Build-a-graph

You have 4 operations to build a graph:

1. Create a new vertex with a label i.
2. Disjoint union of two previously-constructed graphs.
3. Join all vertices labelled i to all labelled j ($i \neq j$).
4. Relabel every vertex labelled i with j.

Example



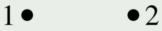
Create vertex with label 1

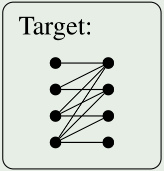
Build-a-graph

You have 4 operations to build a graph:

1. Create a new vertex with a label i.
2. Disjoint union of two previously-constructed graphs.
3. Join all vertices labelled i to all labelled j ($i \neq j$).
4. Relabel every vertex labelled i with j.

Example





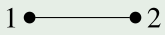
Create vertex with label 2

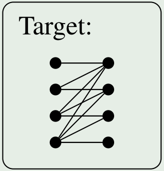
Build-a-graph

You have 4 operations to build a graph:

1. Create a new vertex with a label i.
2. Disjoint union of two previously-constructed graphs.
3. Join all vertices labelled i to all labelled j ($i \neq j$).
4. Relabel every vertex labelled i with j.

Example





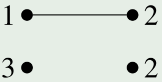
Join labels 1 and 2

Build-a-graph

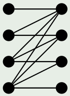
You have 4 operations to build a graph:

1. Create a new vertex with a label i.
2. Disjoint union of two previously-constructed graphs.
3. Join all vertices labelled i to all labelled j ($i \neq j$).
4. Relabel every vertex labelled i with j.

Example



Target:



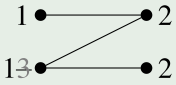
Create vertices with labels 2 and 3 (or use disjoint union)

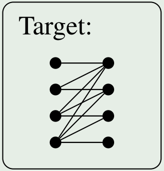
Build-a-graph

You have 4 operations to build a graph:

1. Create a new vertex with a label i.
2. Disjoint union of two previously-constructed graphs.
3. Join all vertices labelled i to all labelled j ($i \neq j$).
4. Relabel every vertex labelled i with j.

Example





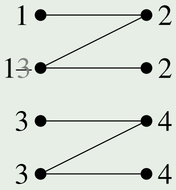
Join labels 2 and 3, and relabel $3 \to 1$

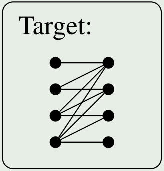
Build-a-graph

You have 4 operations to build a graph:

1. Create a new vertex with a label i.
2. Disjoint union of two previously-constructed graphs.
3. Join all vertices labelled i to all labelled j ($i \neq j$).
4. Relabel every vertex labelled i with j.

Example





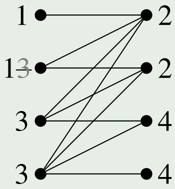
Disjoint union with another copy of the same graph

Build-a-graph

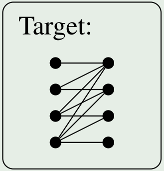
You have 4 operations to build a graph:

1. Create a new vertex with a label i.
2. Disjoint union of two previously-constructed graphs.
3. Join all vertices labelled i to all labelled j ($i \neq j$).
4. Relabel every vertex labelled i with j.

Example



Join labels 2 and 3

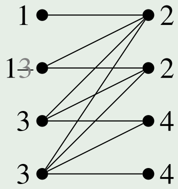


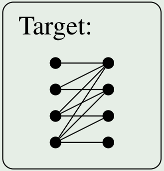
Build-a-graph

You have 4 operations to build a graph:

1. Create a new vertex with a label i.
2. Disjoint union of two previously-constructed graphs.
3. Join all vertices labelled i to all labelled j ($i \neq j$).
4. Relabel every vertex labelled i with j.

Example





Graph built! I used 4 labels.

Build-a-graph

You have 4 operations to build a graph:

1. Create a new vertex with a label i.
2. Disjoint union of two previously-constructed graphs.
3. Join all vertices labelled i to all labelled j ($i \neq j$).
4. Relabel every vertex labelled i with j.

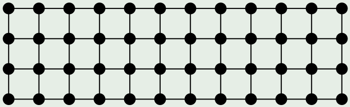
- Clique-width, $cw(G)$ = size of smallest label set needed to build G.
- If $H \leqslant_{\text{ind}} G$, then $cw(H) \leqslant cw(G)$.
- Clique-width of a class $\mathcal{C}$

$$cw(\mathcal{C}) = \max_{G \in \mathcal{C}} cw(G)$$

if this exists.

What has big clique width?

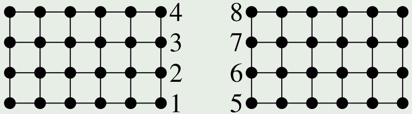
Intuition: Unbounded clique width needs two dimensions.



For fixed k: $cw(k \times n \text{ grid}) = O(k)$

What has big clique width?

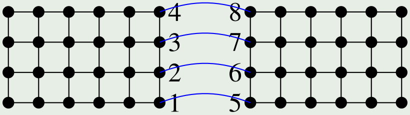
Intuition: Unbounded clique width needs two dimensions.



For fixed k: $cw(k \times n \text{ grid}) = O(k)$

What has big clique width?

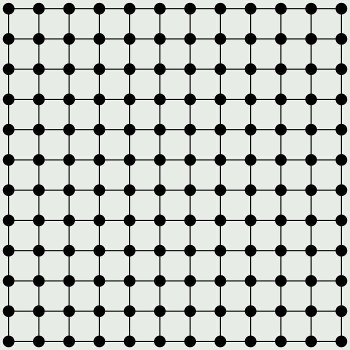
Intuition: Unbounded clique width needs two dimensions.



For fixed k: $cw(k \times n \text{ grid}) = O(k)$

What has big clique width?

Intuition: Unbounded clique width needs two dimensions.



For fixed k: $cw(k \times n \text{ grid}) = O(k)$

$cw(n \times n \text{ grid}) = n + 1$ (Golumbic and Rotics, 1999)

Tree-width, rank-width, clique-width

Theorem (Corneil and Rotics, 2005)

For any graph G,

$$cw(G) \leqslant 3 \cdot 2^{tw(G)}.$$

Note: $cw(K_n) = 2$, but $tw(K_n) = n - 1$.

Theorem (Oum and Seymour, 2006)

For any graph G,

$$rw(G) \leqslant cw(G) \leqslant 2^{rw(G)+1} - 1.$$

Thus:

- Clique-width unbounded implies tree-width unbounded (converse false)
- Rank-width unbounded iff clique-width unbounded

Usefulness of clique-width

Since rank-width and clique-width are related:

Theorem (Courcelle, Makowsky, Rotics, 2000)

Any problem expressible in MSO_1 logic can be solved in linear time on every class of graphs with bounded rank-width.

Usefulness of clique-width

Since rank-width and clique-width are related:

Theorem (Courcelle, Makowsky, Rotics, 2000)

Any problem expressible in MSO_1 logic can be solved in linear time on every class of graphs with bounded clique width.

(In fact, rank-width was only introduced in 2006, so this is more like the original result.)

Is there a ‘grid theorem’ for bounding clique-width in hereditary classes?

Just use the vertex-minor grid theorem?

Hereditary classes are a richer (and arguably more natural) family: every vertex-minor-closed class is hereditary.

The ‘circle graphs’ in the vertex-minor grid theorem contain lots of interesting hereditary classes. Some have bounded clique-width, others don’t.

Clique-width: history to 2011

1993 Courcelle, Engelfriet & Rozenberg: (sort of) introduce clique-width.

1999 Makowsky & Rotics: *split graphs* have unbounded clique-width.

2000 Courcelle, Makowsky & Rotics: MSO_1 metatheorem.

Golumbic & Rotics: *permutation graphs* have unbounded clique-width.

2006 Oum & Seymour introduce rank-width as an approximation for clique-width that can be computed efficiently.

2011 Lozin shows that bipartite permutation graphs and unit interval graphs are minimal classes with unbounded clique-width.

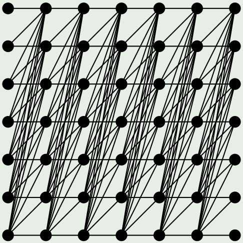
Minimal hereditary classes of unbounded clique-width

Class $\mathcal{C}$ is *minimal (of unbounded clique-width)* if:

- $\mathcal{C}$ has unbounded clique-width, and
- any proper subclass $\mathcal{D} \subsetneq \mathcal{C}$ has bounded clique-width.

Bipartite permutation graphs (Lozin, 2011)

Class comprises all induced subgraphs of grids like the following:



Is there a ‘grid theorem’ for bounding clique-width in hereditary classes?

Is there a ‘grid theorem’ for bounding clique-width in hereditary classes?

Not if you want it to mention just one ‘grid’

Since bipartite permutation graphs and unit interval graphs are both minimal of unbounded clique-width, there are at least two grids…

Is there a ‘grid theorem’ for bounding clique-width in hereditary classes?

Not if you want it to mention just one ‘grid’

Since bipartite permutation graphs and unit interval graphs are both minimal of unbounded clique-width, there are at least two grids…

…but perhaps we could list the minimal classes?

Discovering minimal classes

2011 Lozin: bipartite permutation graphs and unit interval graphs.

Discovering minimal classes

2011 Lozin: bipartite permutation graphs and unit interval graphs.

2015 (published 2021) Atminas, B., Lozin & Stacho:
split permutation graphs and bichain graphs.

Discovering minimal classes

2011 Lozin: bipartite permutation graphs and unit interval graphs.

2015 (published 2021) Atminas, B., Lozin & Stacho:
split permutation graphs and bichain graphs.

2018 Collins, Foniok, Korpelainen, Lozin & Zamaraev:
countably infinite collection of minimal classes

Discovering minimal classes

2011 Lozin: bipartite permutation graphs and unit interval graphs.

2015 (published 2021) Atminas, B., Lozin & Stacho:
split permutation graphs and bichain graphs.

2018 Collins, Foniok, Korpelainen, Lozin & Zamaraev:
countably infinite collection of minimal classes

2022 B. & Cocks: uncountably infinite collection

Discovering minimal classes

2011 Lozin: bipartite permutation graphs and unit interval graphs.

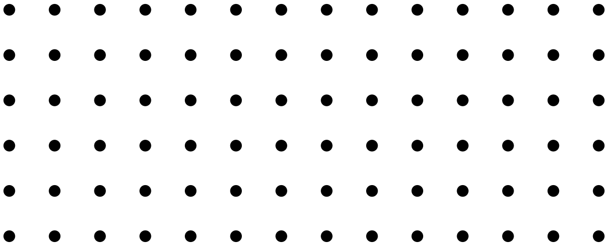
2015 (published 2021) Atminas, B., Lozin & Stacho:
split permutation graphs and bichain graphs.

2018 Collins, Foniok, Korpelainen, Lozin & Zamaraev:
countably infinite collection of minimal classes

2022 B. & Cocks: uncountably infinite collection

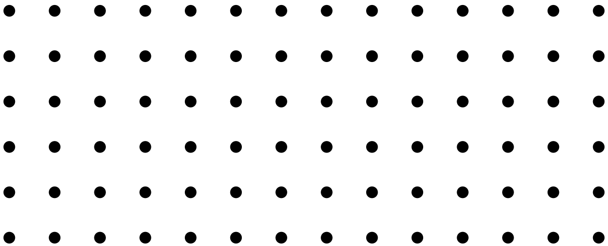
2023 B. & Cocks: General framework for all the above classes.

The framework



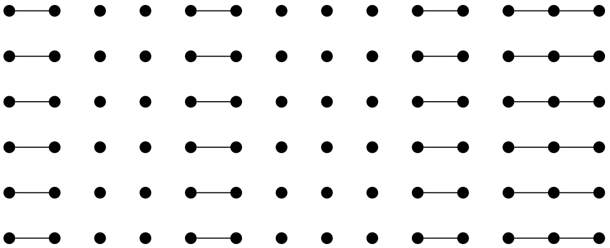
Infinite grid. Insert edges according to a triple of objects (α, β, γ)...

The framework



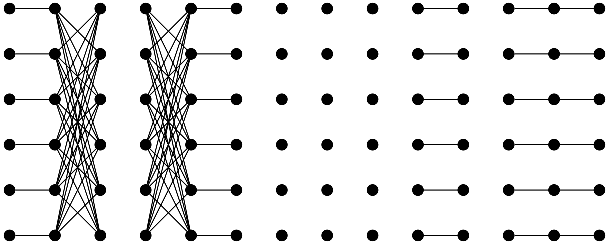
$\alpha \quad 0 \; 1 \; 2 \; 1 \; 0 \; 3 \; 3 \; 2 \; 2 \; 0 \; 2 \; 0 \; 0 \cdots \quad \in \{0, 1, 2, 3\}^*$

The framework



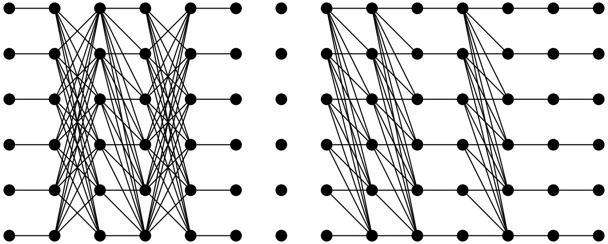
α 0 1 2 1 0 3 3 2 2 0 2 0 0 $\cdots$ $\in \{0, 1, 2, 3\}^*$

The framework



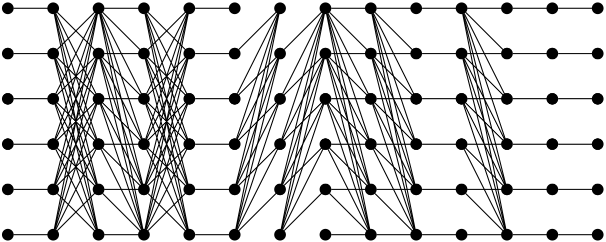
α 0 1 2 1 0 3 3 2 2 0 2 0 0 $\cdots$ $\in \{0, 1, 2, 3\}^*$

The framework



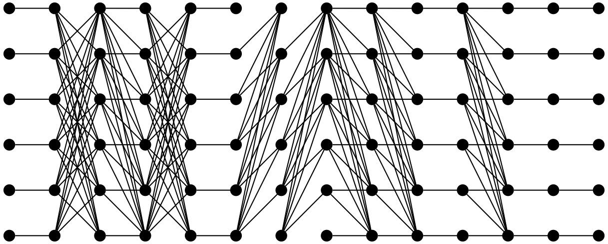
α 0 1 2 1 0 3 3 2 2 0 2 0 0 $\cdots$ $\in \{0, 1, 2, 3\}^*$

The framework



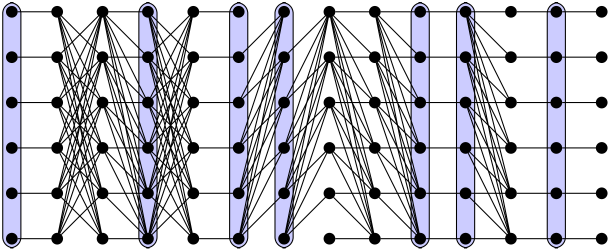
α 0 1 2 1 0 3 3 2 2 0 2 0 0 $\cdots$ $\in \{0, 1, 2, 3\}^*$

The framework



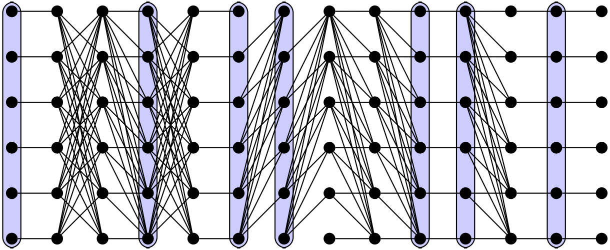
β 1 0 0 1 0 1 1 0 0 1 1 0 1 0 $\cdots$ $\in \{0,1\}^*$

The framework



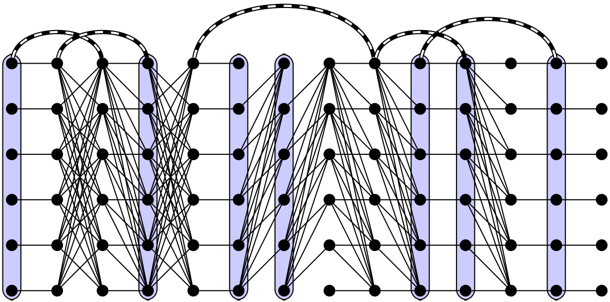
β 1 0 0 1 0 1 1 0 0 1 1 0 1 0 $\cdots$ $\in \{0,1\}^*$

The framework



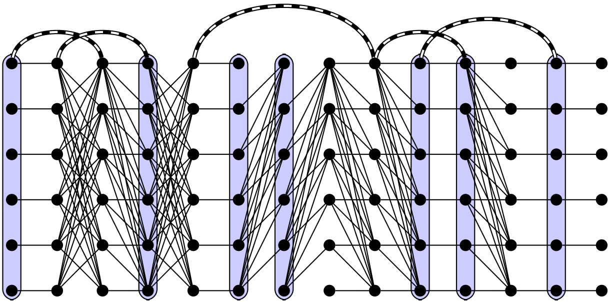
γ $(1,3),(2,4),(5,9),(9,11),(10,13),\dots$ $\subseteq \mathbb{N}\times\mathbb{N}$

The framework



γ $(1,3),(2,4),(5,9),(9,11),(10,13),\ldots$ $\subseteq \mathbb{N}\times\mathbb{N}$

The framework



α		0	1	2	1	0	3	3	2	2	0	2	0	0	$\cdots$	$\in \{0,1,2,3\}^*$
β	1	0	0	1	0	1	1	0	0	1	1	0	1	0	$\cdots$	$\in \{0,1\}^*$

γ $(1,3),(2,4),(5,9),(9,11),(10,13),\ldots$ $\subseteq \mathbb{N} \times \mathbb{N}$

From grids to classes

Each triple $\delta = (\alpha, \beta, \gamma)$ defines an infinite graph H^{δ}, and then

$$\mathcal{G}^{\delta} = \{G \text{ finite} : G \leqslant_{\text{ind}} H^{\delta}\}.$$

From grids to classes

Each triple $\delta = (\alpha, \beta, \gamma)$ defines an infinite graph H^{δ}, and then

$$\mathcal{G}^{\delta} = \{G \text{ finite} : G \leqslant_{\text{ind}} H^{\delta}\}.$$

Theorem (B. & Cocks, 2023)

There exists a parameter N^{δ} relying only on δ such that $cw(\mathcal{G}^{\delta})$ is unbounded if and only if N^{δ} is unbounded.

N^{δ} is unbounded, for example, if α contains infinitely many 2s or 3s.

Uncountably many minimal classes

Not all classes $\mathcal{G}^{\delta}$ of unbounded clique width are minimal, but lots are. One simple family is as follows:

Theorem (B. & Cocks, 2022)

Let $\beta = 00\cdots$, $\gamma = \varnothing$ *and let* α *be any infinite uniformly recurrent word over the alphabet* $\{0, 1, 2, 3\}$ *other than* $00\cdots$.
Then $\mathcal{G}^{(\alpha,\beta,\gamma)}$ *is a minimal hereditary class of unbounded clique width.*

Uniformly recurrent: every factor w of α is guaranteed to appear in α infinitely often, and consecutive occurrences are 'close'.

Uncountably many minimal classes

Not all classes $\mathcal{G}^{\delta}$ of unbounded clique width are minimal, but lots are. One simple family is as follows:

Theorem (B. & Cocks, 2022)

Let $\beta = 00\cdots$, $\gamma = \varnothing$ *and let* α *be any infinite uniformly recurrent word over the alphabet* $\{0, 1, 2, 3\}$ *other than* $00\cdots$.
Then $\mathcal{G}^{(\alpha,\beta,\gamma)}$ *is a minimal hereditary class of unbounded clique width.*

Uniformly recurrent: every factor w of α is guaranteed to appear in α infinitely often, and consecutive occurrences are 'close'.

Sturmian sequences are an uncountably large collection of uniformly recurrent binary sequences $\Rightarrow$ uncountably many minimal classes.

Is there a ‘grid theorem’ for bounding clique-width in hereditary classes?

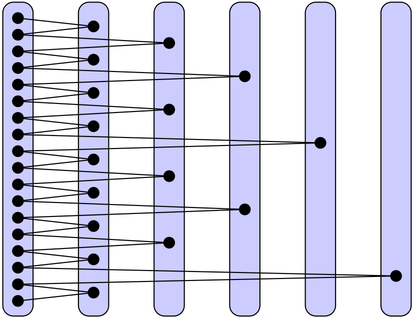
Is there a ‘grid theorem’ for bounding clique-width in hereditary classes?

Well, you can’t list them all…

…but perhaps we are now close to a complete characterisation?

Dragons

'Power graphs' discovered by Lozin, Razgon & Zamaraev (2018), proved minimal by Dawar & Sankaran (2023):



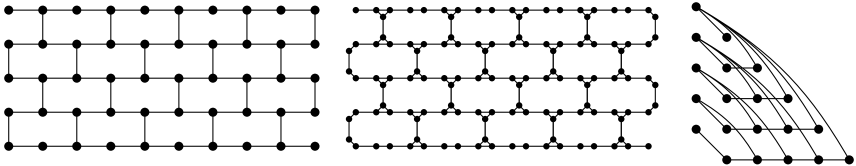
Not part of the previous framework. Also, this class is well-quasi-ordered (if you know what that means).

Sparse dragons

Theorem (Gurski & Wanke, 2000)

If a hereditary class is sparse, then it has unbounded clique-width if and only if it has unbounded tree-width.

Classifying bounded tree-width in sparse graphs is a major topic in itself.



Conjecture (Cocks, 2024)

Sparse hereditary graph classes of unbounded tree-width do not contain a minimal class of unbounded tree-width.

Thanks!

Main references:

- B. & Cocks, *Uncountably many minimal hereditary classes of graphs of unbounded clique-width*, Elec. J. Combin. **29** (2022)
- B. & Cocks, *A framework for minimal hereditary classes of graphs of unbounded clique-width*, SIAM J. Disc. Math. **37** (2023)