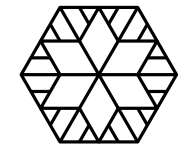
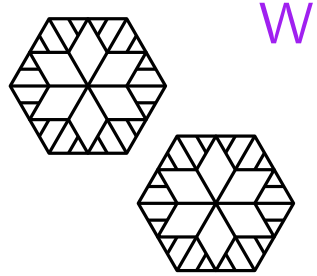
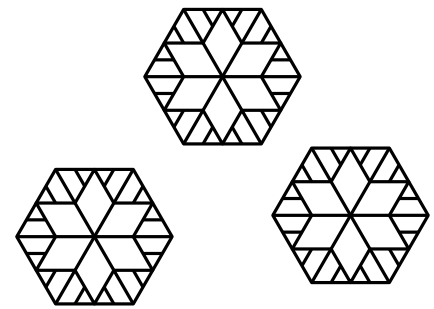
Introduction to **Graph Minor theory** and the **Irrelevant Vertex technique**



Laure Morelle

Winter School





Please **ask questions** during the talk!

- if you (or I) forgot a definition
- if you don't understand something
- if you're curious about something

Please **ask questions** during the talk!

- if you (or I) forgot a definition
- if you don't understand something
- if you're curious about something

Because:

- there are no stupid questions
- if you don't understand something, you are certainly not the only one and the others will be extremely grateful to you
- I don't bite

Please **ask questions** during the talk!

- if you (or I) forgot a definition
- if you don't understand something
- if you're curious about something

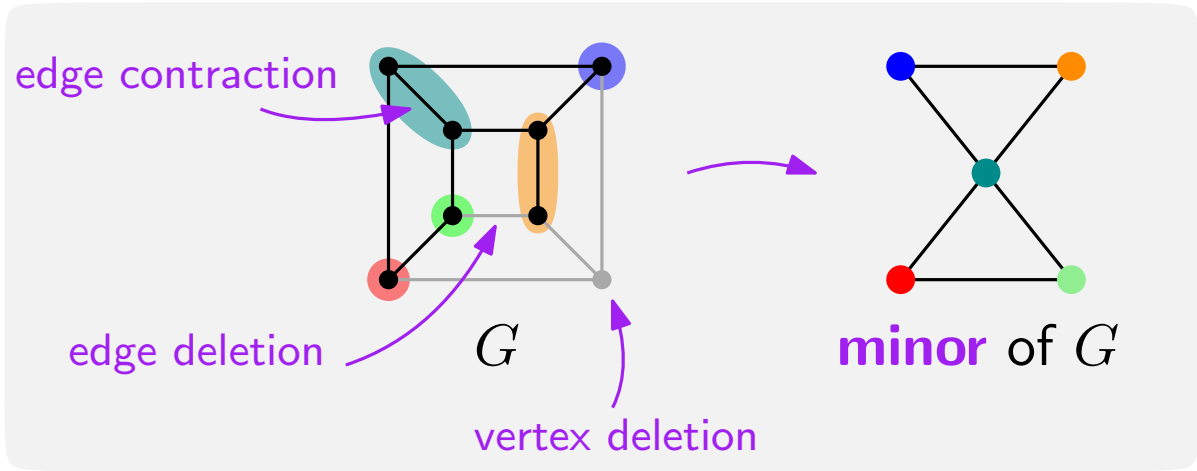
Because:

- there are no stupid questions
- if you don't understand something, you are certainly not the only one and the others will be extremely grateful to you
- I don't bite

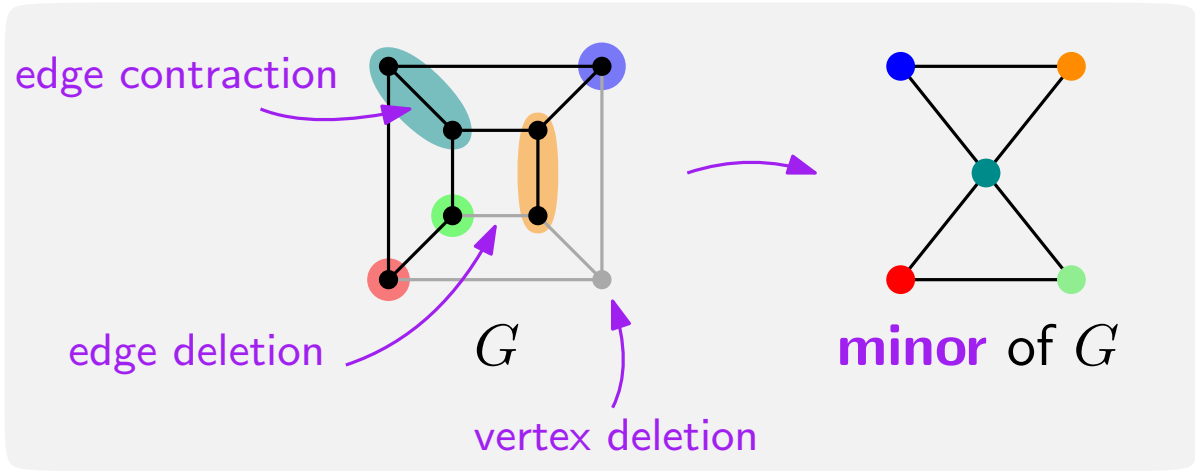
Also:

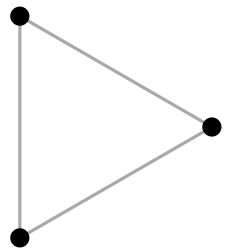
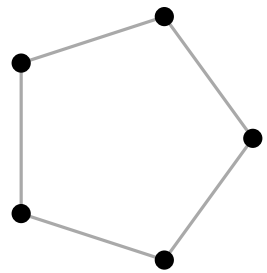
Take a paper and a pen please.

Minors

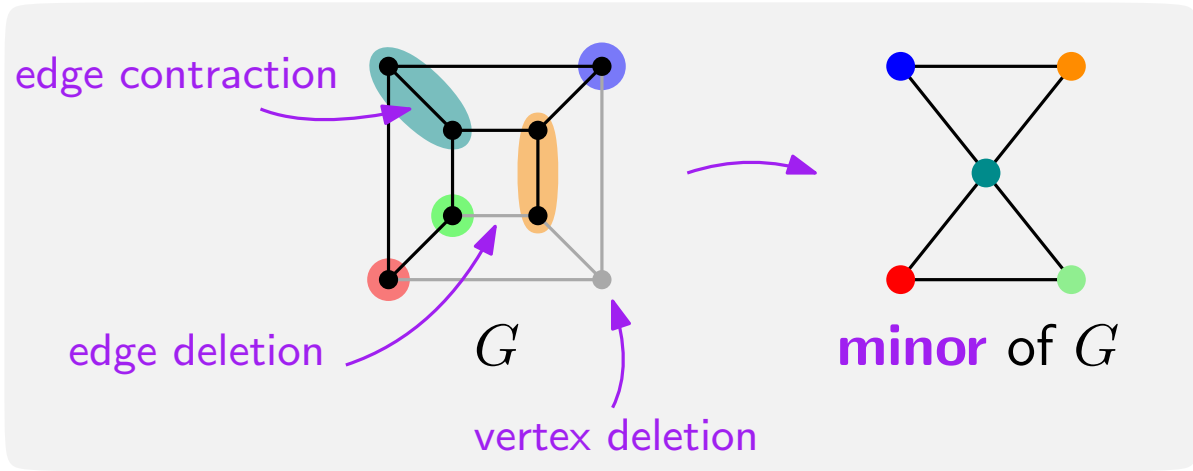

edge contraction
edge deletion
G
vertex deletion
minor of G

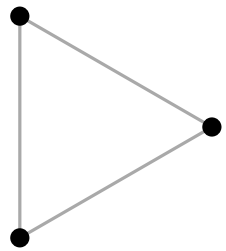
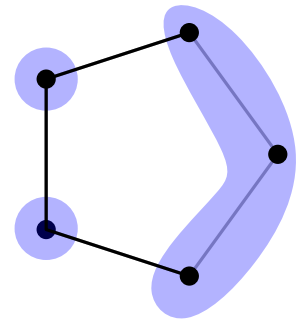
Minors



Is  a minor of  ?

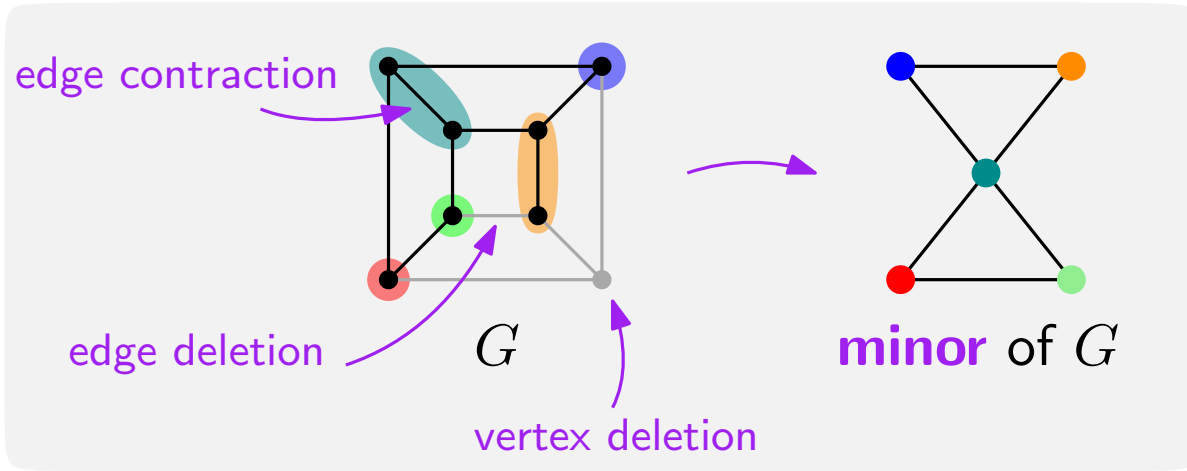
Minors

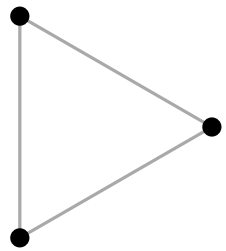
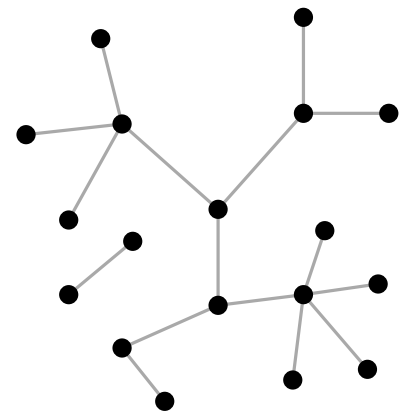


Is  a minor of  ? yes

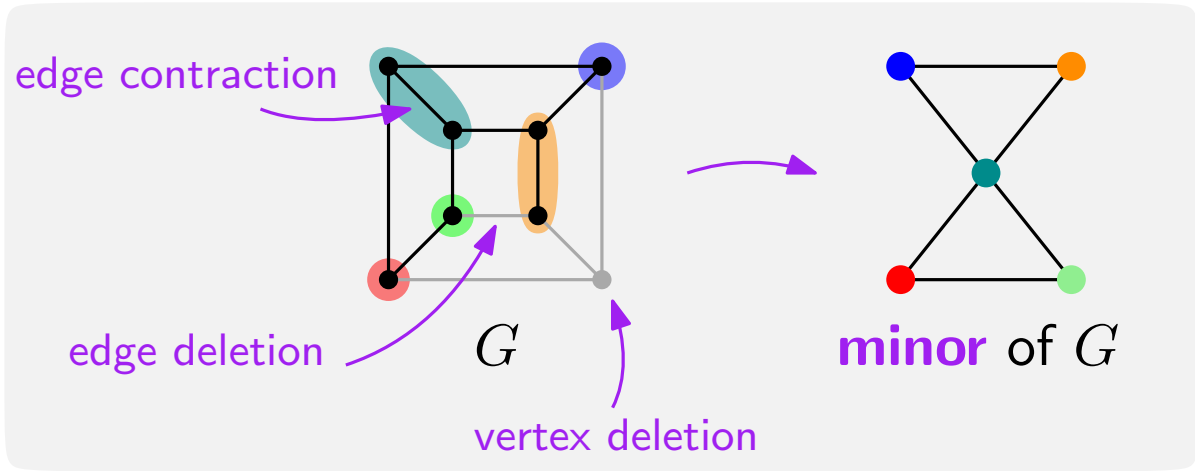
K_3 is a minor of every cycle.

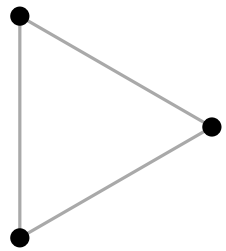
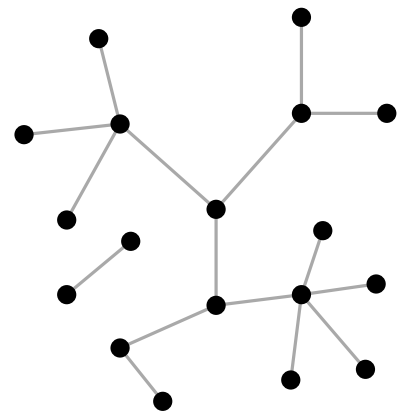
Minors



Is  a minor of  ?

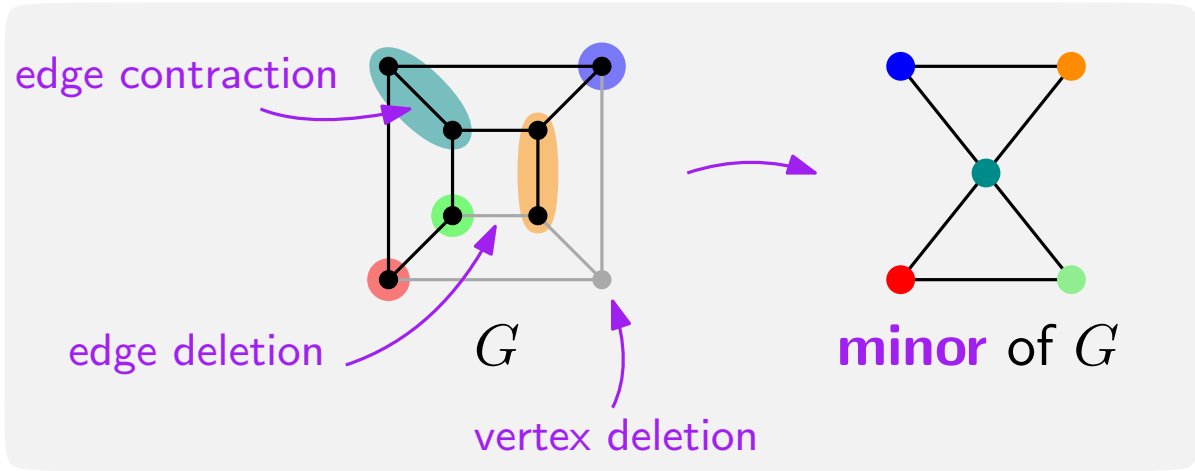
Minors

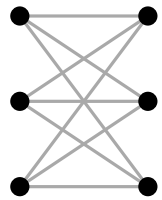
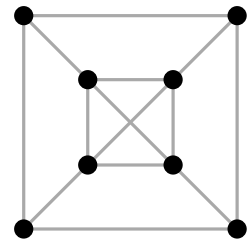


Is  a minor of  ? No

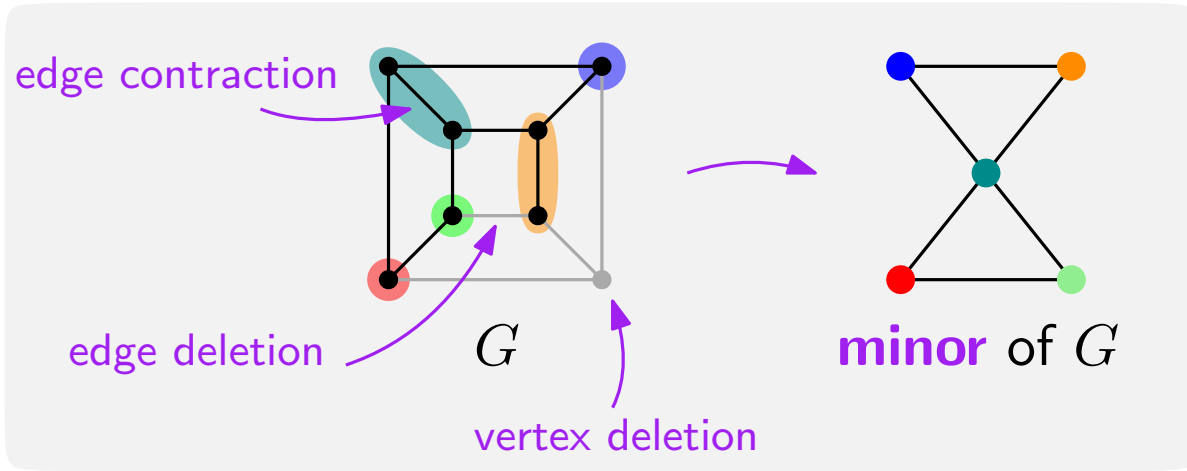
forests $=$ K_3-minor-free graphs

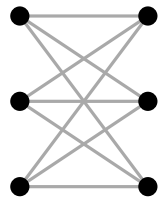
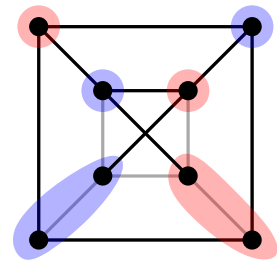
Minors



Is  a minor of  ?

Minors



Is  a minor of  ? yes

Minor-closed graph classes

A graph class $\mathcal{H}$ is **minor-closed** if all minors of graphs in $\mathcal{H}$ are also in $\mathcal{H}$.

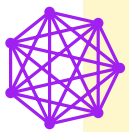
i.e. after deleting a vertex, deleting an edge, or contracting an edge, we are still in $\mathcal{H}$

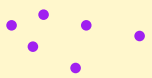
Minor-closed graph classes

A graph class $\mathcal{H}$ is **minor-closed** if all minors of graphs in $\mathcal{H}$ are also in $\mathcal{H}$.

i.e. after deleting a vertex, deleting an edge, or contracting an edge, we are still in $\mathcal{H}$

Are those minor-closed?

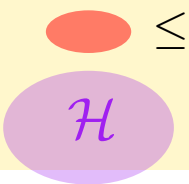




- edgeless graphs
- cliques
- trees

- forests
- bipartite graphs

- planar graphs
- $\mathcal{G}_{\mathcal{H},k}$ = graphs that belong to the minor-closed graph class $\mathcal{H}$ after deleting $\leq k$ vertices

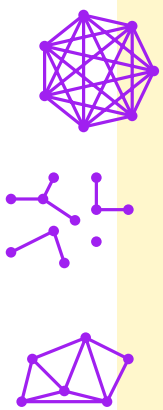


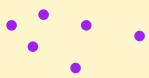
Minor-closed graph classes

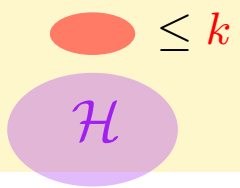
A graph class $\mathcal{H}$ is **minor-closed** if all minors of graphs in $\mathcal{H}$ are also in $\mathcal{H}$.

i.e. after deleting a vertex, deleting an edge, or contracting an edge, we are still in $\mathcal{H}$

Are those minor-closed?



- edgeless graphs  yes
- cliques
- trees 
- forests
- bipartite graphs 
- planar graphs
- $\mathcal{G}_{\mathcal{H},k}$ = graphs that belong to the minor-closed graph class $\mathcal{H}$ after deleting $\leq k$ vertices

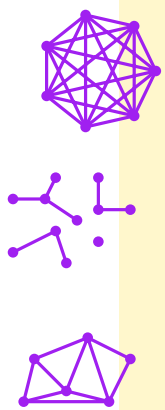


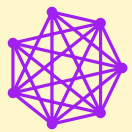
Minor-closed graph classes

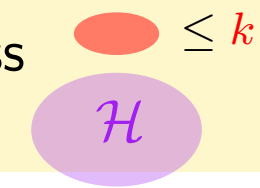
A graph class $\mathcal{H}$ is **minor-closed** if all minors of graphs in $\mathcal{H}$ are also in $\mathcal{H}$.

i.e. after deleting a vertex, deleting an edge, or contracting an edge, we are still in $\mathcal{H}$

Are those minor-closed?



- edgeless graphs  yes
- cliques no 
- trees 
- forests
- bipartite graphs 
- planar graphs
- $\mathcal{G}_{\mathcal{H},k}$ = graphs that belong to the minor-closed graph class $\mathcal{H}$ after deleting $\leq k$ vertices

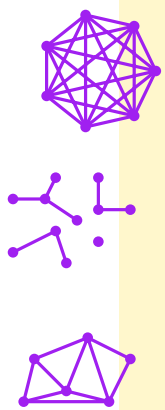


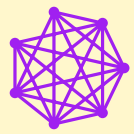
Minor-closed graph classes

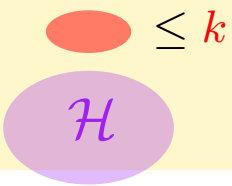
A graph class $\mathcal{H}$ is **minor-closed** if all minors of graphs in $\mathcal{H}$ are also in $\mathcal{H}$.

i.e. after deleting a vertex, deleting an edge, or contracting an edge, we are still in $\mathcal{H}$

Are those minor-closed?



- edgeless graphs  yes
- cliques no 
- trees  no 
- forests
- bipartite graphs 
- planar graphs
- $\mathcal{G}_{\mathcal{H},k}$ = graphs that belong to the minor-closed graph class $\mathcal{H}$ after deleting $\leq k$ vertices

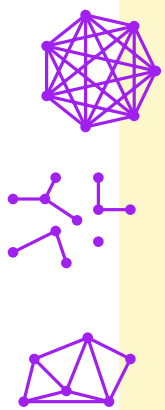


Minor-closed graph classes

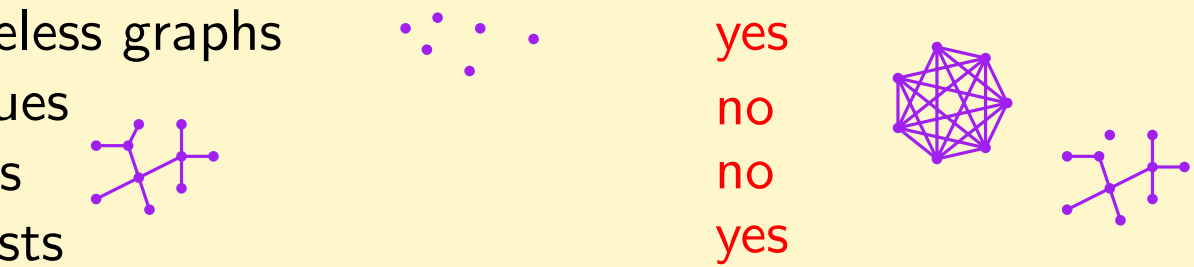
A graph class $\mathcal{H}$ is **minor-closed** if all minors of graphs in $\mathcal{H}$ are also in $\mathcal{H}$.

i.e. after deleting a vertex, deleting an edge, or contracting an edge, we are still in $\mathcal{H}$

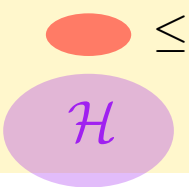
Are those minor-closed?



- edgeless graphs — yes
- cliques — no
- trees — no
- forests — yes
- bipartite graphs
- planar graphs



- $\mathcal{G}_{\mathcal{H},k}$ = graphs that belong to the minor-closed graph class $\mathcal{H}$ after deleting $\leq k$ vertices

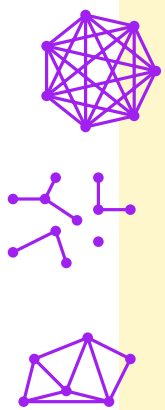


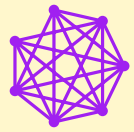
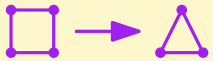
Minor-closed graph classes

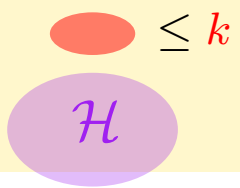
A graph class $\mathcal{H}$ is **minor-closed** if all minors of graphs in $\mathcal{H}$ are also in $\mathcal{H}$.

i.e. after deleting a vertex, deleting an edge, or contracting an edge, we are still in $\mathcal{H}$

Are those minor-closed?



- edgeless graphs  yes
- cliques no 
- trees  no 
- forests yes
- bipartite graphs  no 
- planar graphs
- $\mathcal{G}_{\mathcal{H},k}$ = graphs that belong to the minor-closed graph class $\mathcal{H}$ after deleting $\leq k$ vertices

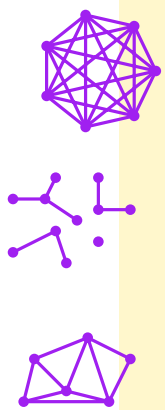


Minor-closed graph classes

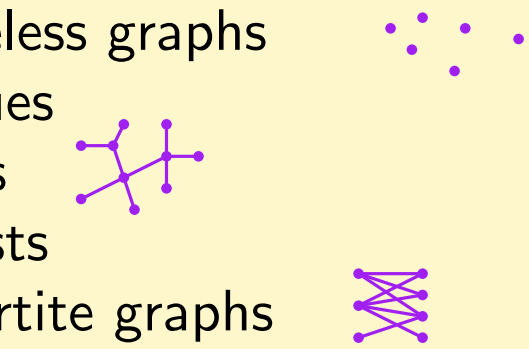
A graph class $\mathcal{H}$ is **minor-closed** if all minors of graphs in $\mathcal{H}$ are also in $\mathcal{H}$.

i.e. after deleting a vertex, deleting an edge, or contracting an edge, we are still in $\mathcal{H}$

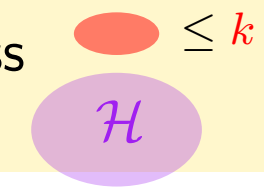
Are those minor-closed?



- edgeless graphs — yes
- cliques — no
- trees — no
- forests — yes
- bipartite graphs — no
- planar graphs — yes



- $\mathcal{G}_{\mathcal{H},k}$ = graphs that belong to the minor-closed graph class $\mathcal{H}$ after deleting $\leq k$ vertices

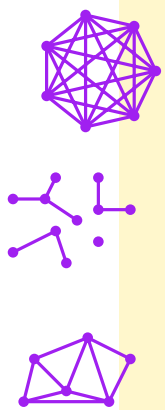


Minor-closed graph classes

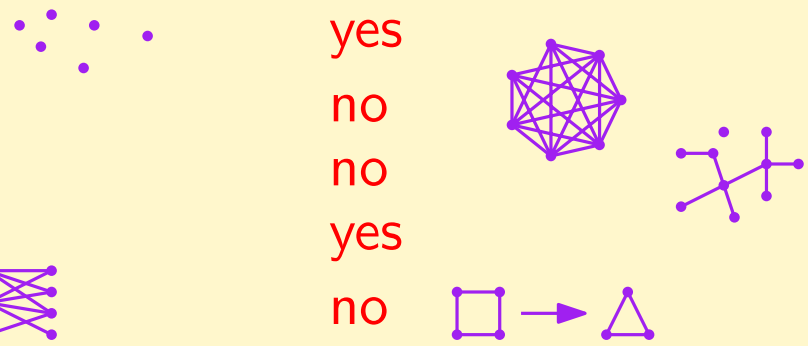
A graph class $\mathcal{H}$ is **minor-closed** if all minors of graphs in $\mathcal{H}$ are also in $\mathcal{H}$.

i.e. after deleting a vertex, deleting an edge, or contracting an edge, we are still in $\mathcal{H}$

Are those minor-closed?



- edgeless graphs — yes
- cliques — no
- trees — no
- forests — yes
- bipartite graphs — no
- planar graphs — yes
- $\mathcal{G}_{\mathcal{H},k}$ = graphs that belong to the minor-closed graph class $\mathcal{H}$ after deleting $\leq k$ vertices — yes

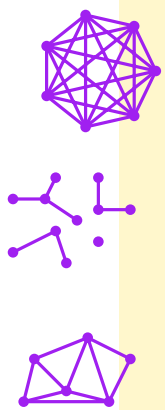


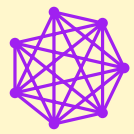
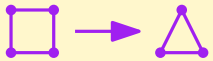
Minor-closed graph classes

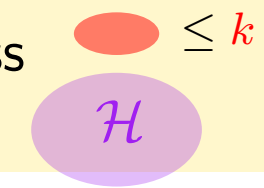
A graph class $\mathcal{H}$ is **minor-closed** if all minors of graphs in $\mathcal{H}$ are also in $\mathcal{H}$.

i.e. after deleting a vertex, deleting an edge, or contracting an edge, we are still in $\mathcal{H}$

Are those minor-closed?

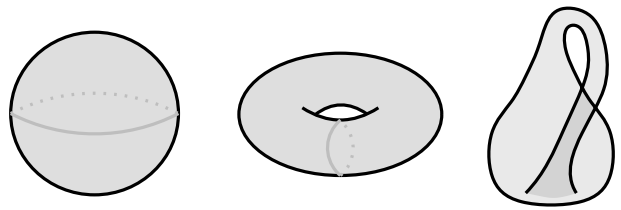


- edgeless graphs  yes
- cliques no 
- trees  no 
- forests yes
- bipartite graphs  no 
- planar graphs yes
- $\mathcal{G}_{\mathcal{H},k}$ = graphs that belong to the minor-closed graph class $\mathcal{H}$ after deleting $\leq k$ vertices yes



Other minor-closed graph classes:

- outerplanar graphs
- graphs embeddable on a surface



Why Graph Minor theory?

Why Graph Minor theory?

because of

the Graph Minors Series of Robertson and Seymour

23 papers, over more than 20 years (1983-2012)

Introduce important results and most of the techniques used in Graph Minor theory.

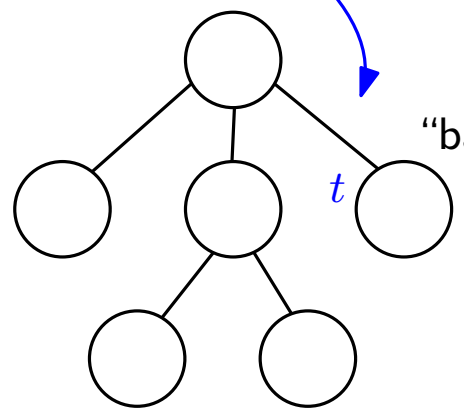
Treewidth,
the **Courcelle's theorem**,
the **Grid Exclusion theorem**,
and the **Graph Minor structure theorem**

[Graph Minors II & V & XVI], [Courcelle, 1990]

[Graph Minors II] (Re)introduction of the **treewidth**

[Graph Minors II] (Re)introduction of the **treewidth**

Tree decomposition $(T, \{B_t\}_{t \in V(T)})$ of G



"bag" of t

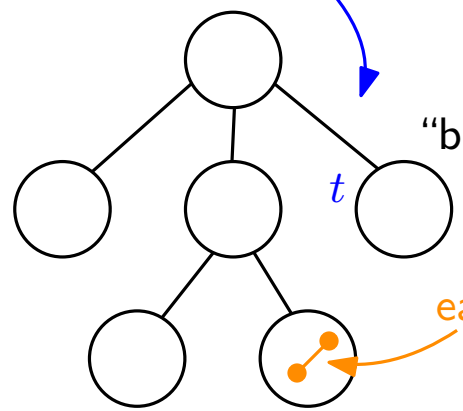
t

$B_t \subseteq V(G)$

$\bigcup_{t \in V(T)} B_t = V(G)$

[Graph Minors II] (Re)introduction of the **treewidth**

Tree decomposition $(T, \{B_t\}_{t \in V(T)})$ of G

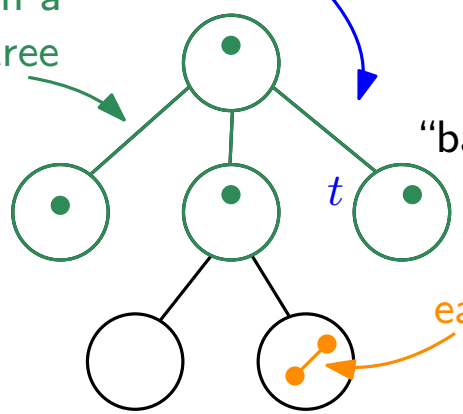


$\bigcup_{t \in V(T)} B_t = V(G)$

[Graph Minors II] (Re)introduction of the **treewidth**

Tree decomposition $(T, \{B_t\}_{t \in V(T)})$ of G

each vertex is in a connected subtree



"bag" of t

t

$B_t \subseteq V(G)$

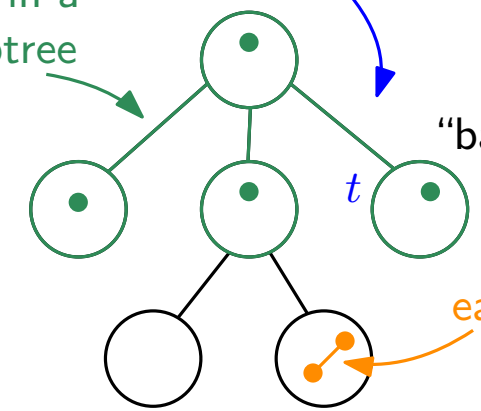
each edge is in a bag

$\bigcup_{t \in V(T)} B_t = V(G)$

[Graph Minors II] (Re)introduction of the **treewidth**

Tree decomposition $(T, \{B_t\}_{t \in V(T)})$ of G

each vertex is in a connected subtree



width $= \max_t |B_t| - 1$

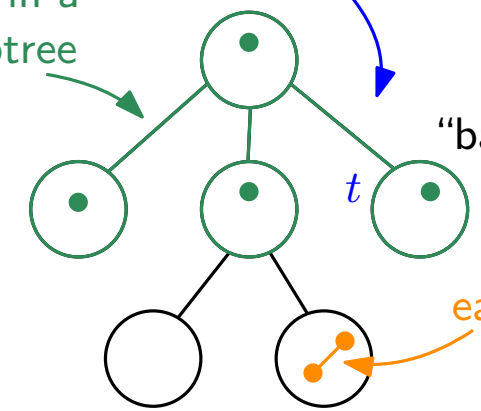
$\bigcup_{t \in V(T)} B_t = V(G)$

Treewidth of G = minimum width of a tree decomposition of G

[Graph Minors II] (Re)introduction of the **treewidth**

Tree decomposition $(T, \{B_t\}_{t \in V(T)})$ of G

each vertex is in a connected subtree



width $= \max_t |B_t| - 1$

$\bigcup_{t \in V(T)} B_t = V(G)$

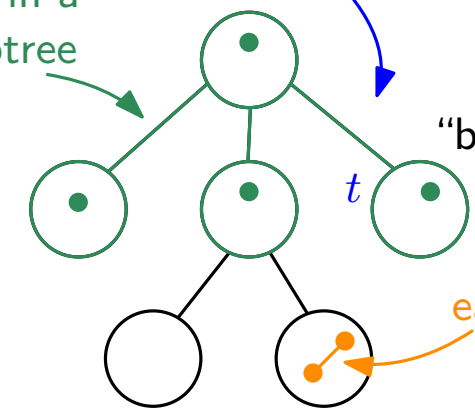
Treewidth of G = minimum width of a tree decomposition of G

treewidth?

[Graph Minors II] (Re)introduction of the **treewidth**

Tree decomposition $(T, \{B_t\}_{t \in V(T)})$ of G

each vertex is in a connected subtree



"bag" of t

$B_t \subseteq V(G)$

each edge is in a bag

width $= \max_t |B_t| - 1$

$\bigcup_{t \in V(T)} B_t = V(G)$

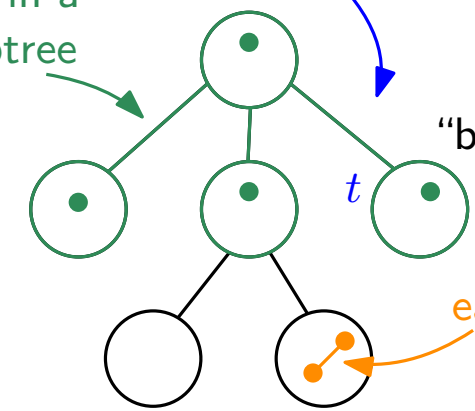
Treewidth of G = minimum width of a tree decomposition of G

treewidth?

[Graph Minors II] (Re)introduction of the **treewidth**

Tree decomposition $(T, \{B_t\}_{t \in V(T)})$ of G

each vertex is in a connected subtree

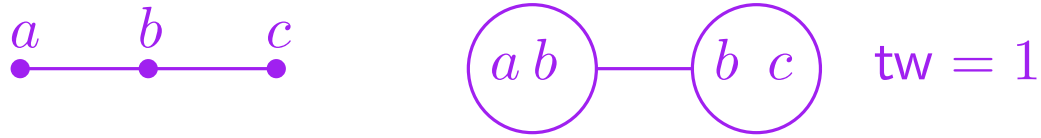


width $= \max_t |B_t| - 1$

$\bigcup_{t \in V(T)} B_t = V(G)$

Treewidth of G = minimum width of a tree decomposition of G

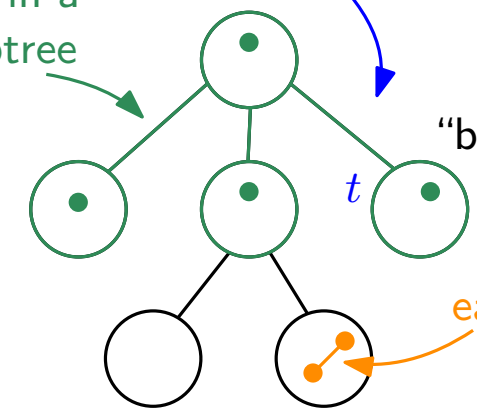
treewidth?



[Graph Minors II] (Re)introduction of the **treewidth**

Tree decomposition $(T, \{B_t\}_{t \in V(T)})$ of G

each vertex is in a connected subtree

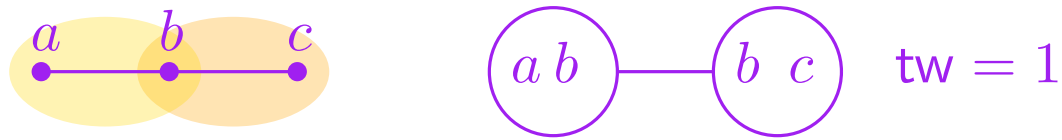


width $= \max_t |B_t| - 1$

$\bigcup_{t \in V(T)} B_t = V(G)$

Treewidth of G = minimum width of a tree decomposition of G

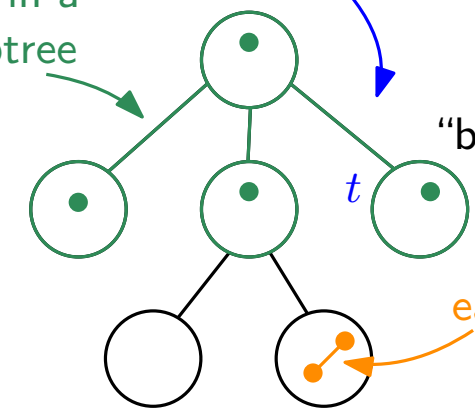
treewidth?



[Graph Minors II] (Re)introduction of the **treewidth**

Tree decomposition $(T, \{B_t\}_{t \in V(T)})$ of G

each vertex is in a connected subtree



"bag" of t

$B_t \subseteq V(G)$

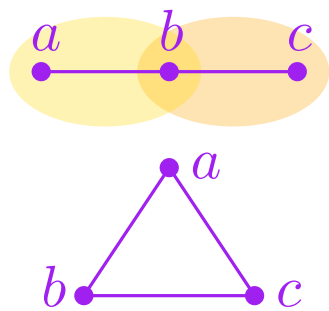
each edge is in a bag

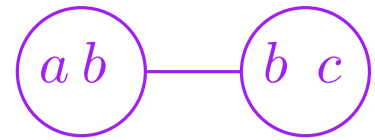
width $= \max_t |B_t| - 1$

$\bigcup_{t \in V(T)} B_t = V(G)$

Treewidth of G = minimum width of a tree decomposition of G

treewidth?



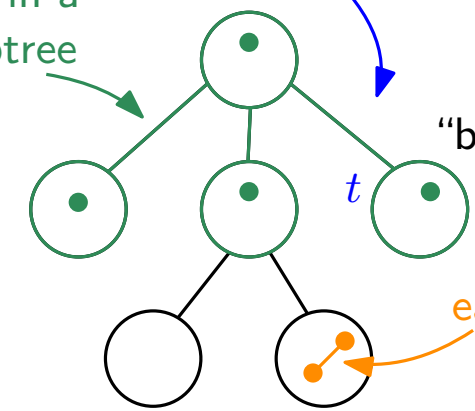


tw $= 1$

[Graph Minors II] (Re)introduction of the **treewidth**

Tree decomposition $(T, \{B_t\}_{t \in V(T)})$ of G

each vertex is in a connected subtree



"bag" of t

$B_t \subseteq V(G)$

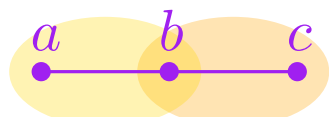
each edge is in a bag

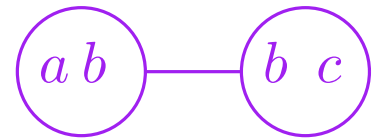
width $= \max_t |B_t| - 1$

$\bigcup_{t \in V(T)} B_t = V(G)$

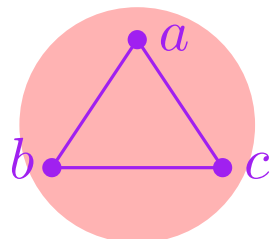
Treewidth of G = minimum width of a tree decomposition of G

treewidth?





tw $= 1$

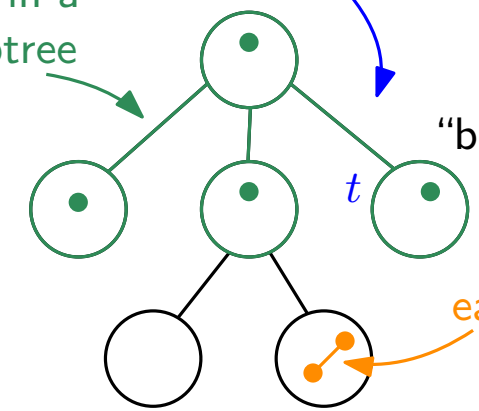


tw $= 2$

[Graph Minors II] (Re)introduction of the **treewidth**

Tree decomposition $(T, \{B_t\}_{t \in V(T)})$ of G

each vertex is in a connected subtree

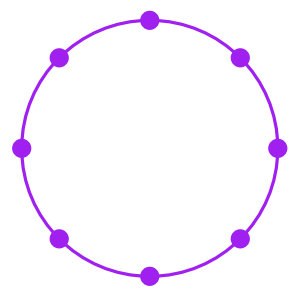


width $= \max_t |B_t| - 1$

$\bigcup_{t \in V(T)} B_t = V(G)$

Treewidth of G = minimum width of a tree decomposition of G

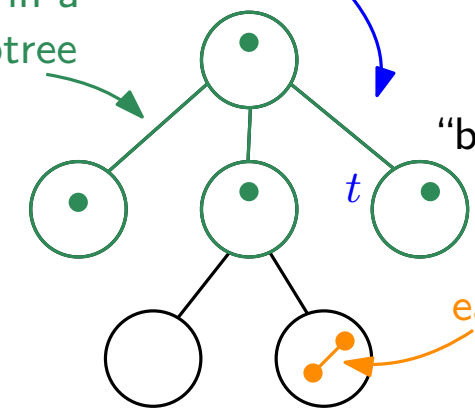
treewidth of a path? of a cycle?



[Graph Minors II] (Re)introduction of the **treewidth**

Tree decomposition $(T, \{B_t\}_{t \in V(T)})$ of G

each vertex is in a connected subtree

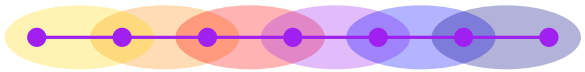


width $= \max_t |B_t| - 1$

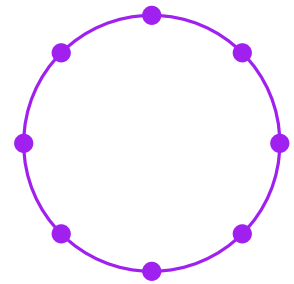
$\bigcup_{t \in V(T)} B_t = V(G)$

Treewidth of G = minimum width of a tree decomposition of G

treewidth of a path? of a cycle?



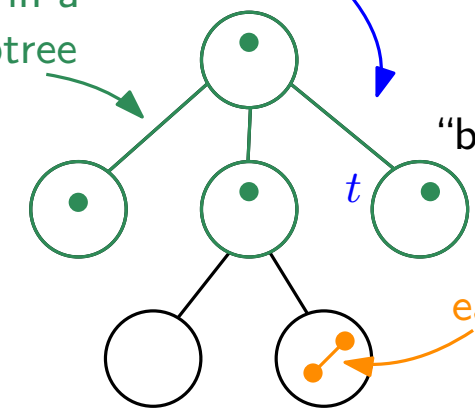
$\text{tw} = 1$



[Graph Minors II] (Re)introduction of the **treewidth**

Tree decomposition $(T, \{B_t\}_{t \in V(T)})$ of G

each vertex is in a connected subtree



"bag" of t

$B_t \subseteq V(G)$

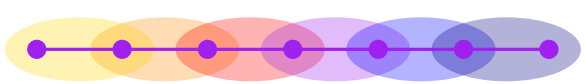
each edge is in a bag

width $= \max_t |B_t| - 1$

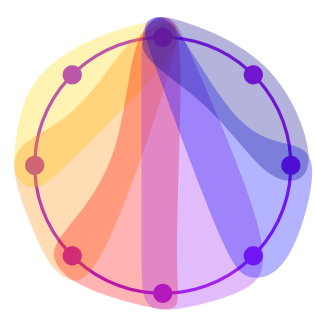
$\bigcup_{t \in V(T)} B_t = V(G)$

Treewidth of G = minimum width of a tree decomposition of G

treewidth of a path? of a cycle?

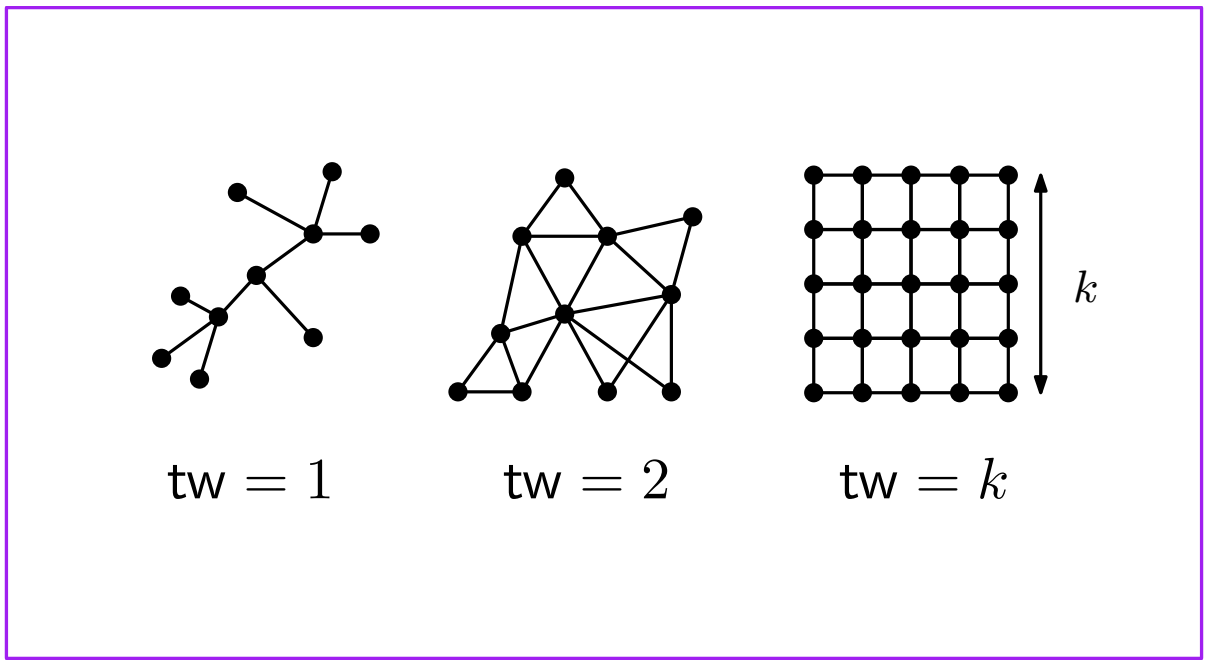


tw $= 1$



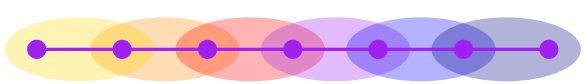
tw $= 2$

[Graph Minors II] (Re)introduction of the **treewidth**

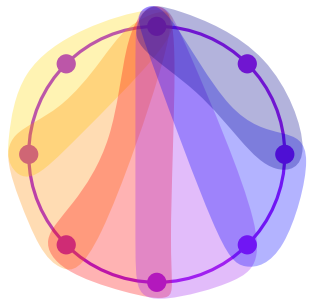


Treewidth of G = minimum width of a tree decomposition of G

treewidth of a path? of a cycle?



tw = 1



tw = 2

Uses of treewidth

Uses of treewidth

Using dynamic programming, most problems are solvable in linear time on graphs of bounded treewidth

Uses of treewidth

Using dynamic programming, most problems are solvable in linear time on graphs of bounded treewidth

[Courcelle, '90]

Every problem expressible in **CMSO logic** is solvable in time $f(k) \cdot n$ on graphs of treewidth $\leq k$.

Uses of treewidth

Using dynamic programming, most problems are solvable in linear time on graphs of bounded treewidth

[Courcelle, '90]

Every problem expressible in **CMSO logic** is solvable in time $f(k) \cdot n$ on graphs of treewidth $\leq k$.

First-order (FO) logic:

Variables: vertex v, edge e

$\exists, \forall, \wedge, \vee, \neg, =, \mathsf{inc}(\cdot,\cdot), \mathsf{adj}(\cdot,\cdot)$

$\mathsf{inc}(v,e) = e$ is incident to v, $\mathsf{adj}(v_1,v_2) = v_1$ is adjacent to v_2

Uses of treewidth

Using dynamic programming, most problems are solvable in linear time on graphs of bounded treewidth

[Courcelle, '90]

Every problem expressible in **CMSO logic** is solvable in time $f(k) \cdot n$ on graphs of treewidth $\leq k$.

First-order (FO) logic:

Variables: vertex v, edge e

$\exists, \forall, \wedge, \vee, \neg, =, \mathsf{inc}(\cdot,\cdot), \mathsf{adj}(\cdot,\cdot)$

$\mathsf{inc}(v,e) = e$ is incident to v, $\mathsf{adj}(v_1,v_2) = v_1$ is adjacent to v_2

Property of containing a triangle as a subgraph:

$\exists v_1, \exists v_2, \exists v_3, (\mathsf{adj}(v_1,v_2) \wedge (\mathsf{adj}(v_2,v_3)) \wedge (\mathsf{adj}(v_1,v_3))$

Uses of treewidth

Using dynamic programming, most problems are solvable in linear time on graphs of bounded treewidth

[Courcelle, '90]

Every problem expressible in **CMSO logic** is solvable in time $f(k) \cdot n$ on graphs of treewidth $\leq k$.

First-order (FO) logic:

Variables: vertex v, edge e

$\exists, \forall, \wedge, \vee, \neg, =, \mathsf{inc}(\cdot,\cdot), \mathsf{adj}(\cdot,\cdot)$

$\mathsf{inc}(v,e) = e$ is incident to v, $\mathsf{adj}(v_1,v_2) = v_1$ is adjacent to v_2

How to express the property of having a vertex cover of size 2?

Property of containing a triangle as a subgraph:

$\exists v_1, \exists v_2, \exists v_3, (\mathsf{adj}(v_1,v_2) \wedge (\mathsf{adj}(v_2,v_3)) \wedge (\mathsf{adj}(v_1,v_3))$

Uses of treewidth

Using dynamic programming, most problems are solvable in linear time on graphs of bounded treewidth

[Courcelle, '90]

Every problem expressible in **CMSO logic** is solvable in time $f(k) \cdot n$ on graphs of treewidth $\leq k$.

First-order (FO) logic:

Variables: vertex v, edge e

$\exists, \forall, \wedge, \vee, \neg, =, \mathsf{inc}(\cdot,\cdot), \mathsf{adj}(\cdot,\cdot)$

How to express the property of having a vertex cover of size 2?

$\exists v_1, \exists v_2, \forall e, (\mathsf{inc}(v_1, e) \vee (\mathsf{inc}(v_2, e))$

$\mathsf{inc}(v, e) = e$ is incident to v, $\mathsf{adj}(v_1, v_2) = v_1$ is adjacent to v_2

Property of containing a triangle as a subgraph:

$\exists v_1, \exists v_2, \exists v_3, (\mathsf{adj}(v_1, v_2) \wedge (\mathsf{adj}(v_2, v_3)) \wedge (\mathsf{adj}(v_1, v_3))$

Uses of treewidth

Using dynamic programming, most problems are solvable in linear time on graphs of bounded treewidth

[Courcelle, '90]

Every problem expressible in **CMSO logic** is solvable in time $f(k) \cdot n$ on graphs of treewidth $\leq k$.

First-order (FO) logic:

Variables: vertex v, edge e

$\exists, \forall, \wedge, \vee, \neg, =, \mathsf{inc}(\cdot,\cdot), \mathsf{adj}(\cdot,\cdot)$

How to express the property of having a vertex cover of size 2?

$\exists v_1, \exists v_2, \forall e, (\mathsf{inc}(v_1, e) \vee (\mathsf{inc}(v_2, e))$

$\mathsf{inc}(v, e) = e$ is incident to v, $\mathsf{adj}(v_1, v_2) = v_1$ is adjacent to v_2

Monadic Second-Order (MSO_2) logic:

extension of FO logic with:

New variables: vertex set V, edge set E

New relation: $\in$

Uses of treewidth

Using dynamic programming, most problems are solvable in linear time on graphs of bounded treewidth

[Courcelle, '90]

Every problem expressible in **CMSO logic** is solvable in time $f(k) \cdot n$ on graphs of treewidth $\leq k$.

First-order (FO) logic:

Variables: vertex v, edge e

$\exists, \forall, \wedge, \vee, \neg, =, \mathsf{inc}(\cdot,\cdot), \mathsf{adj}(\cdot,\cdot)$

How to express the property of having a vertex cover of size 2?

$\exists v_1, \exists v_2, \forall e, (\mathsf{inc}(v_1, e) \vee (\mathsf{inc}(v_2, e))$

$\mathsf{inc}(v, e) = e$ is incident to v, $\mathsf{adj}(v_1, v_2) = v_1$ is adjacent to v_2

Monadic Second-Order (MSO_2) logic:

extension of FO logic with:

New variables: vertex set V, edge set E

New relation: $\in$

How to express 2-colorability?

Uses of treewidth

Using dynamic programming, most problems are solvable in linear time on graphs of bounded treewidth

[Courcelle, '90]

Every problem expressible in **CMSO logic** is solvable in time $f(k) \cdot n$ on graphs of treewidth $\leq k$.

First-order (FO) logic:

Variables: vertex v, edge e

$\exists, \forall, \wedge, \vee, \neg, =, \mathsf{inc}(\cdot,\cdot), \mathsf{adj}(\cdot,\cdot)$

How to express the property of having a vertex cover of size 2?

$\exists v_1, \exists v_2, \forall e, (\mathsf{inc}(v_1, e) \vee (\mathsf{inc}(v_2, e))$

$\mathsf{inc}(v, e) = e$ is incident to v, $\mathsf{adj}(v_1, v_2) = v_1$ is adjacent to v_2

Monadic Second-Order (MSO_2) logic:

extension of FO logic with:

New variables: vertex set V, edge set E

New relation: $\in$

How to express 2-colorability?

$\exists V, \forall v_1, \forall v_2, \neg(\mathsf{adj}(v_1, v_2) \wedge$
$((v_1 \in V \wedge v_2 \in V) \vee (\neg v_1 \in V \wedge \neg v_2 \in V)))$

Uses of treewidth

Using dynamic programming, most problems are solvable in linear time on graphs of bounded treewidth

[Courcelle, '90]

Every problem expressible in **CMSO logic** is solvable in time $f(k) \cdot n$ on graphs of treewidth $\leq k$.

First-order (FO) logic:

Variables: vertex v, edge e

$\exists, \forall, \wedge, \vee, \neg, =, \mathsf{inc}(\cdot,\cdot), \mathsf{adj}(\cdot,\cdot)$

How to express the property of having a vertex cover of size 2?

$\exists v_1, \exists v_2, \forall e, (\mathsf{inc}(v_1, e) \vee (\mathsf{inc}(v_2, e))$

$\mathsf{inc}(v, e) = e$ is incident to v, $\mathsf{adj}(v_1, v_2) = v_1$ is adjacent to v_2

Monadic Second-Order (MSO_2) logic:

extension of FO logic with:

New variables: vertex set V, edge set E

New relation: $\in$

How to express 2-colorability?

$\exists V, \forall v_1, \forall v_2, \neg(\mathsf{adj}(v_1, v_2) \wedge$
$((v_1 \in V \wedge v_2 \in V) \vee (\neg v_1 \in V \wedge \neg v_2 \in V)))$

How to express connectivity?

Uses of treewidth

Using dynamic programming, most problems are solvable in linear time on graphs of bounded treewidth

[Courcelle, '90]

Every problem expressible in **CMSO logic** is solvable in time $f(k) \cdot n$ on graphs of treewidth $\leq k$.

First-order (FO) logic:

Variables: vertex v, edge e

$\exists, \forall, \wedge, \vee, \neg, =, \mathsf{inc}(\cdot,\cdot), \mathsf{adj}(\cdot,\cdot)$

How to express the property of having a vertex cover of size 2?

$\exists v_1, \exists v_2, \forall e, (\mathsf{inc}(v_1, e) \vee (\mathsf{inc}(v_2, e))$

$\mathsf{inc}(v, e) = e$ is incident to v, $\mathsf{adj}(v_1, v_2) = v_1$ is adjacent to v_2

Monadic Second-Order (MSO_2) logic:

extension of FO logic with:

New variables: vertex set V, edge set E

New relation: $\in$

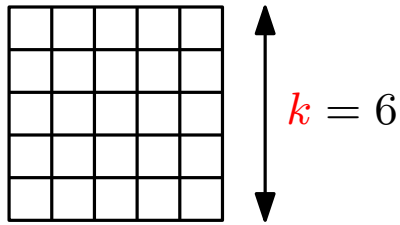
How to express 2-colorability?

$\exists V, \forall v_1, \forall v_2, \neg(\mathsf{adj}(v_1, v_2) \wedge$
$((v_1 \in V \wedge v_2 \in V) \vee (\neg v_1 \in V \wedge \neg v_2 \in V)))$

How to express connectivity?

Counting Monadic Second-Order (CMSO) logic: extension of MSO_2 with $|\cdot| = q \mod r$ $(r \geq 2)$

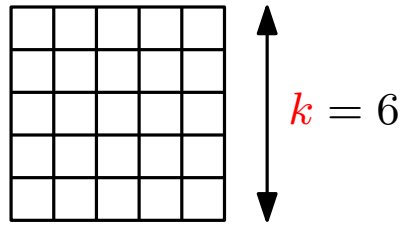
[Graph Minors V] Grid Exclusion theorem



Grid Exclusion theorem:

$\exists g$ s.t., if a graph does not contain a grid of height k as a minor, then it has treewidth $\leq g(k)$.

[Graph Minors V] Grid Exclusion theorem



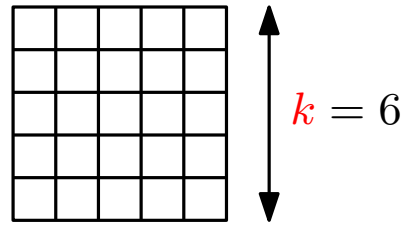
Grid Exclusion theorem:

$\exists g$ s.t., if a graph does not contain a grid of height k as a minor, then it has treewidth $\leq g(k)$.

An application: Win/Win strategy for Feedback Vertex Set

Input: G, k.
Question: Is there a vertex set S of size $\leq k$ such that $G - S$ is a forest?

[Graph Minors V] Grid Exclusion theorem



Grid Exclusion theorem:

$\exists g$ s.t., if a graph does not contain a grid of height k as a minor, then it has treewidth $\leq g(k)$.

An application: Win/Win strategy for Feedback Vertex Set

- If G contains a grid of heigth $2k$ as a minor
  $\rightarrow$ no-instance.
- Otherwise, G has treewidth $\leq g(k)$
  $\rightarrow$ conclude using Courcelle's theorem.

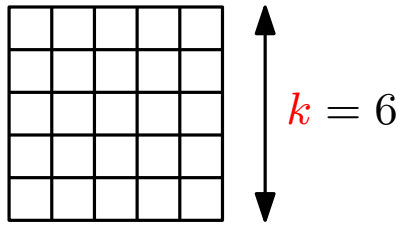
Input: G, k.
Question: Is there a vertex set S of size $\leq k$ such that $G - S$ is a forest?

$\rightarrow$ Solve FVS in FPT-time parameterized by k.

Courcelle's theorem

Every problem expressible in CMSO logic is solvable in time $f(k) \cdot n$ on graphs of treewidth $\leq k$.

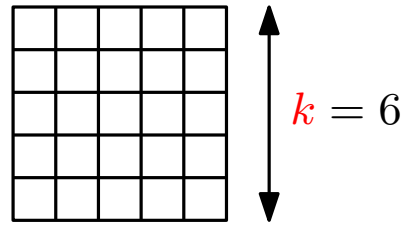
[Graph Minors V] Excluding a planar graph



Grid Exclusion theorem:

$\exists g$ s.t., if a graph does not contain a grid of height k as a minor, then it has treewidth $\leq g(k)$.

[Graph Minors V] Excluding a planar graph

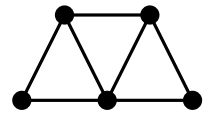


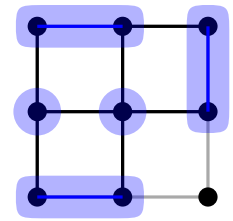
Grid Exclusion theorem:

$\exists g$ s.t., if a graph does not contain a grid of height k as a minor, then it has treewidth $\leq g(k)$.

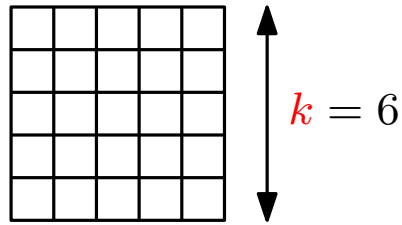
Observation:

Every planar graph is a minor of a big enough grid.





[Graph Minors V] Excluding a planar graph

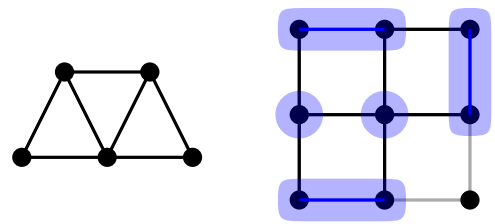


Grid Exclusion theorem:

$\exists g$ s.t., if a graph does not contain a grid of height k as a minor, then it has treewidth $\leq g(k)$.

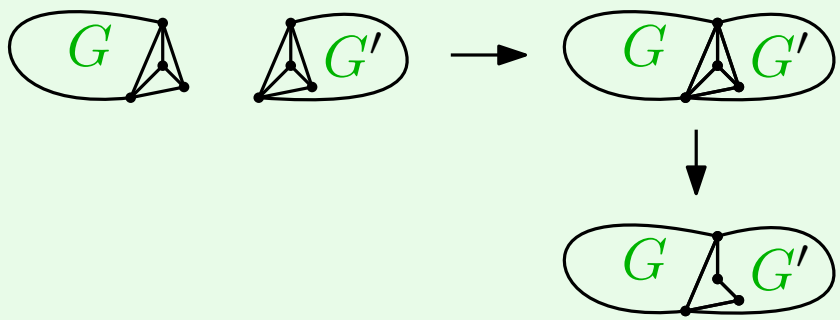
Observation:

Every planar graph is a minor of a big enough grid.

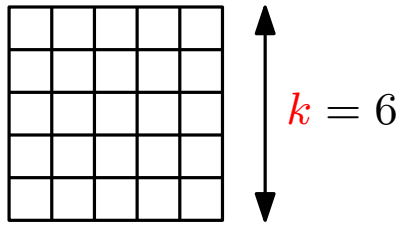


Equivalent definition of the treewidth:

A **clique-sum** of G and G':
- identify a clique of G with a clique of G'
- can delete some edges of the clique



[Graph Minors V] Excluding a planar graph

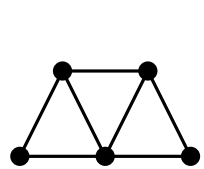


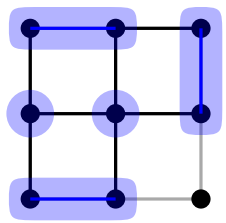
Grid Exclusion theorem:

$\exists g$ s.t., if a graph does not contain a grid of height k as a minor, then it has treewidth $\leq g(k)$.

Observation:

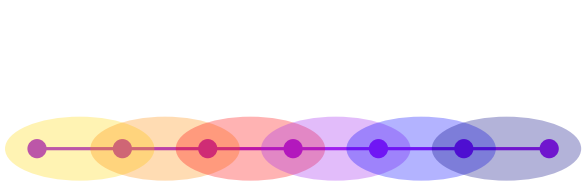
Every planar graph is a minor of a big enough grid.

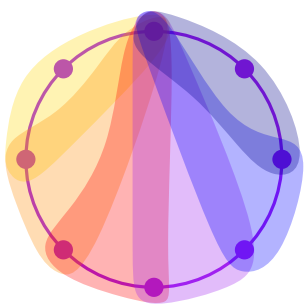




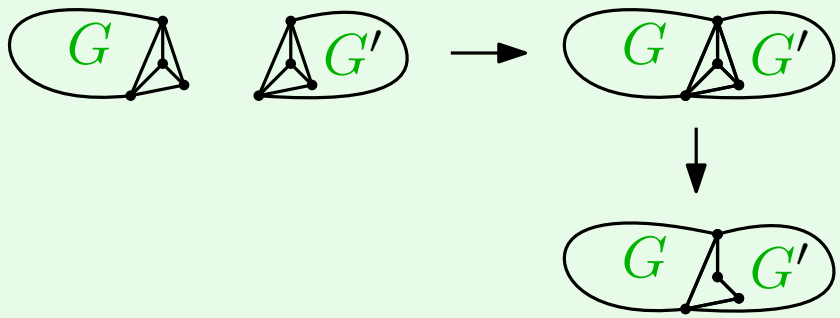
Equivalent definition of the treewidth:

A graph has treewidth $\leq k$ if it can be formed as a clique-sum of graphs with $\leq k+1$ vertices.

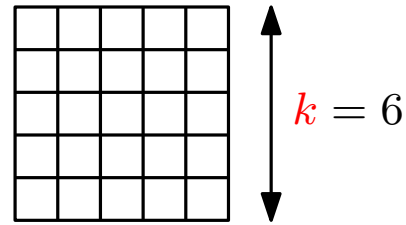




A **clique-sum** of G and G':
- identify a clique of G with a clique of G'
- can delete some edges of the clique



[Graph Minors V] Excluding a planar graph

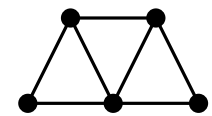


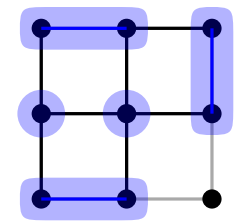
Grid Exclusion theorem:

$\exists g$ s.t., if a graph does not contain a grid of height k as a minor, then it has treewidth $\leq g(k)$.

Observation:

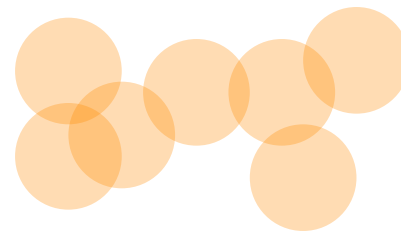
Every planar graph is a minor of a big enough grid.





Equivalent definition of the treewidth:

A graph has treewidth $\leq k$ if it can be formed as a clique-sum of graphs with $\leq k + 1$ vertices.



Structure theorem (planar case)

If a graph G excludes a planar graph H as a minor, then it is a clique-sum of graphs of size $\leq c_H$.

[Graph Minors XVI] Excluding a non-planar graph

Graph Minor Structure theorem

If a graph G excludes a graph H as a minor, then it looks like this:

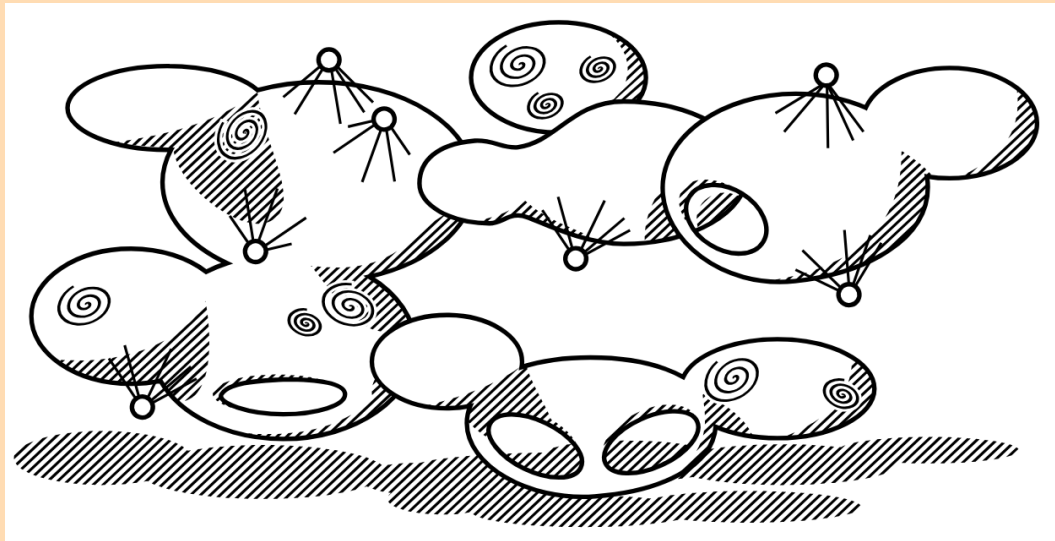


figure by Felix Reidl

[Graph Minors XVI] Excluding a non-planar graph

Graph Minor Structure theorem

If a graph G excludes a graph H as a minor, then it looks like this:

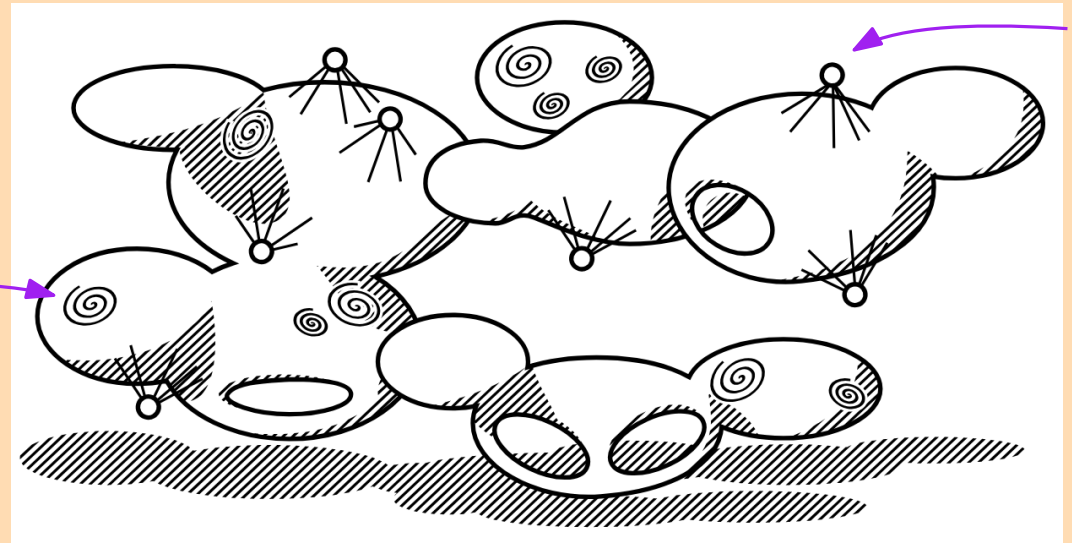


figure by Felix Reidl

i.e. G is a clique-sum of graphs that, after removing $\leq a_H$ apices, are embeddable in a surface of genus $\leq g_H$, aside from $\leq b_H$ regions of "width" $\leq w_H$ called vortices.

[Graph Minors XVI] Excluding a non-planar graph

Graph Minor Structure theorem

If a graph G excludes a graph H as a minor, then it looks like this:

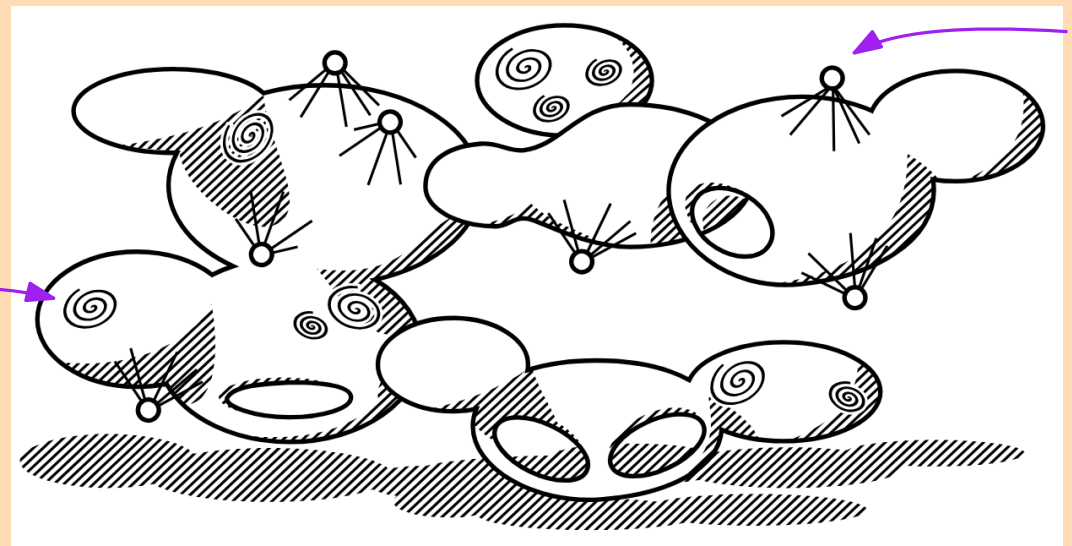


figure by Felix Reidl

i.e. G is a clique-sum of graphs that, after removing $\leq a_H$ apices, are embeddable in a surface of genus $\leq g_H$, aside from $\leq b_H$ regions of "width" $\leq w_H$ called vortices.

If know how to solve a problem on planar graphs, can then try to solve the problem on bounded genus graphs, and then on minor-closed graph classes.

The **Irrelevant Vertex technique**

and the **Unique Linkage theorem**
and the **Flat Wall theorem**

[Graph Minors XIII & XXI]

[Graph Minors XIII] The **Irrelevant Vertex technique**

[Graph Minors XIII] The **Irrelevant Vertex technique**

A very powerful technique for problems related to minor-closed graph classes...

[Graph Minors XIII] The **Irrelevant Vertex technique**

A very powerful technique for problems related to minor-closed graph classes...

> *irrelevant vertex technique* to a standard algorithmic paradigm for solving such problems. For an indicative (while not exhaustive) list of papers that made use of this technique, see [7, 8, 31, 104, 136, 226, 228, 232, 239, 241, 244, 307–310, 318, 329, 349, 366, 369, 378, 379, 385, 386, 389, 391, 393, 400–403, 406, 407, 470, 489, 491, 541, 564–567, 594]. A wide family of problems where this technique has found

[stolen from Giannos Stamoulis' PhD thesis]

[Graph Minors XIII] The **Irrelevant Vertex technique**

A very powerful technique for problems related to minor-closed graph classes...

irrelevant vertex technique to a standard algorithmic paradigm for solving such problems. For an indicative (while not exhaustive) list of papers that made use of this technique, see [7, 8, 31, 104, 136, 226, 228, 232, 239, 241, 244, 307–310, 318, 329, 349, 366, 369, 378, 379, 385, 386, 389, 391, 393, 400–403, 406, 407, 470, 489, 491, 541, 564–567, 594]. A wide family of problems where this technique has found

[stolen from Giannos Stamoulis' PhD thesis]

... but very quite horrible to use.

cause of 40+ pages papers

[Graph Minors XIII] The **Irrelevant Vertex technique**

A very powerful technique for problems related to minor-closed graph classes...

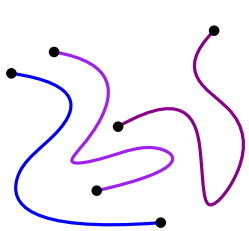
irrelevant vertex technique to a standard algorithmic paradigm for solving such problems. For an indicative (while not exhaustive) list of papers that made use of this technique, see [7, 8, 31, 104, 136, 226, 228, 232, 239, 241, 244, 307–310, 318, 329, 349, 366, 369, 378, 379, 385, 386, 389, 391, 393, 400–403, 406, 407, 470, 489, 491, 541, 564–567, 594]. A wide family of problems where this technique has found

[stolen from Giannos Stamoulis' PhD thesis]

... but very quite horrible to use.

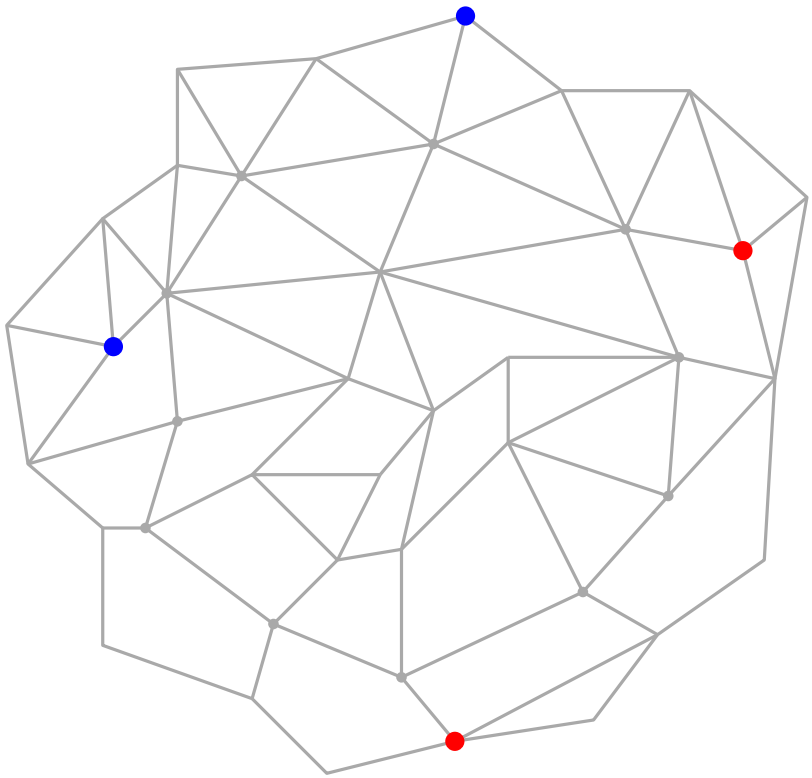
cause of 40+ pages papers

Example of the k-Disjoint Paths problem (kDPP)

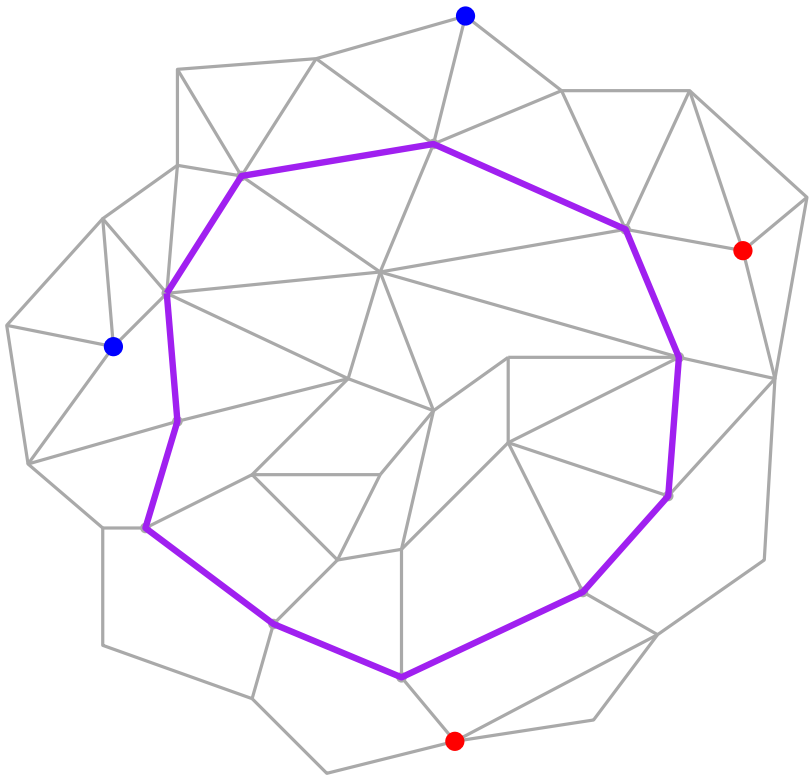


Input: A graph G and terminals $(s_1, t_1), \ldots, (s_k, t_k)$.
Questions: Is there k disjoint paths $P_1, \ldots, P_k$ s.t. P_i is a s_i-t_i path for $1 \leq i \leq k$?

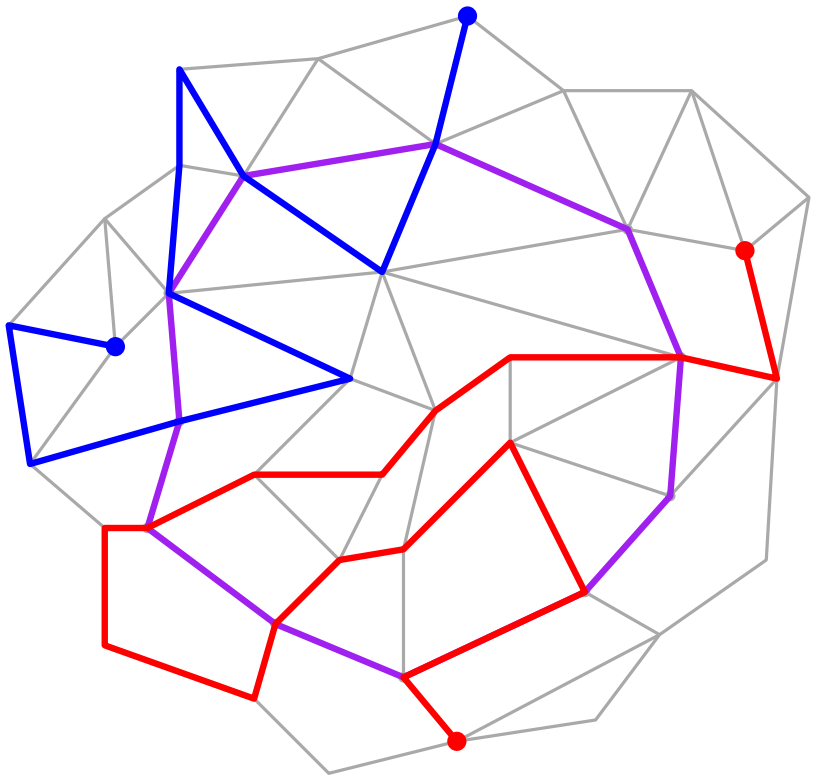
Easy case: 2DPP on planar graphs



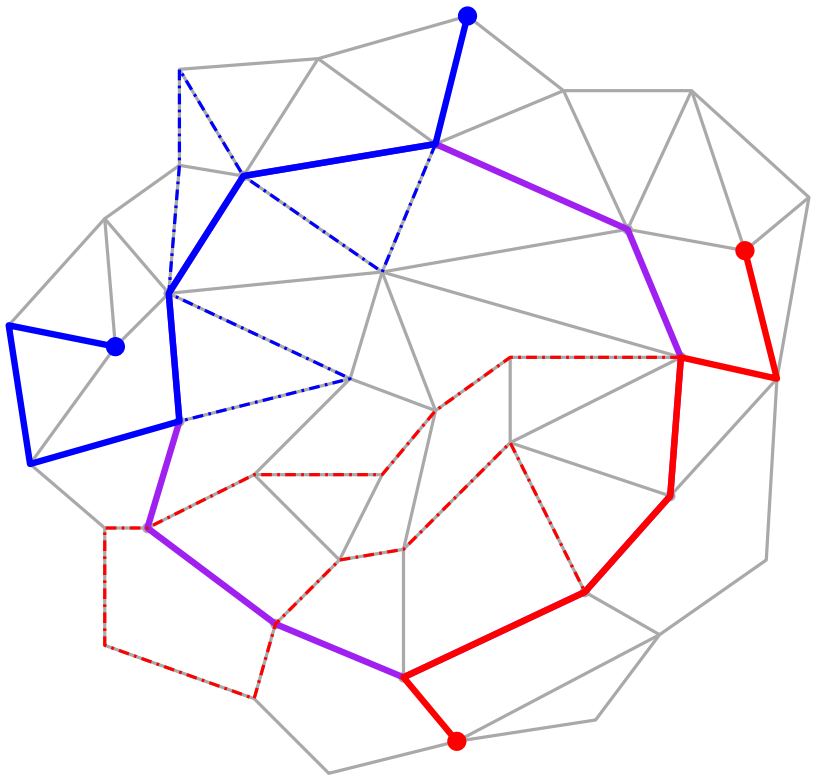
Easy case: 2DPP on planar graphs



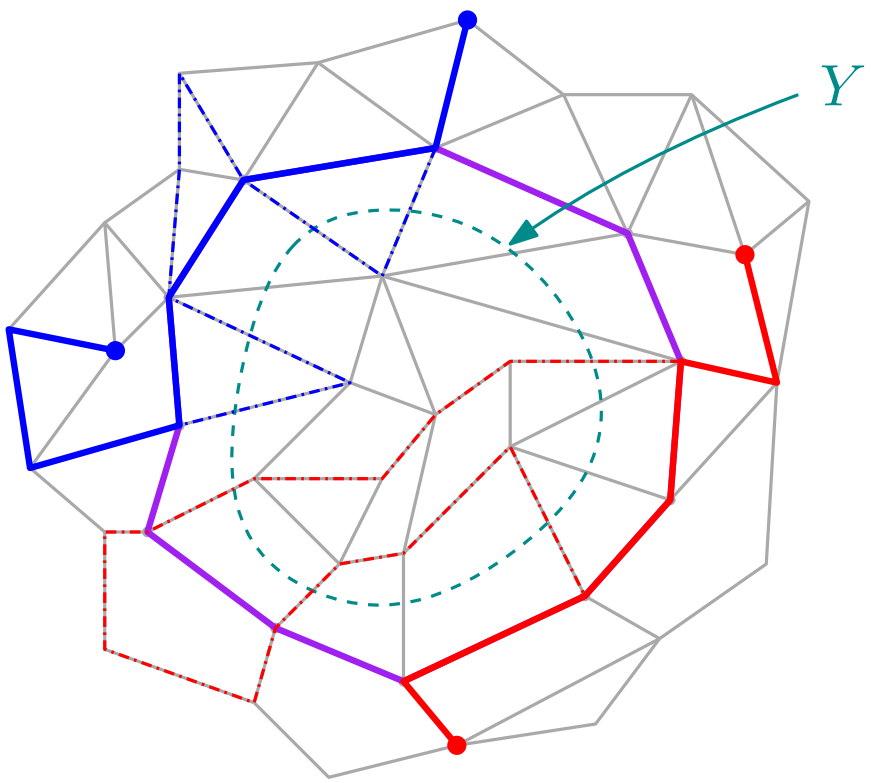
Easy case: 2DPP on planar graphs



Easy case: 2DPP on planar graphs



Easy case: 2DPP on planar graphs



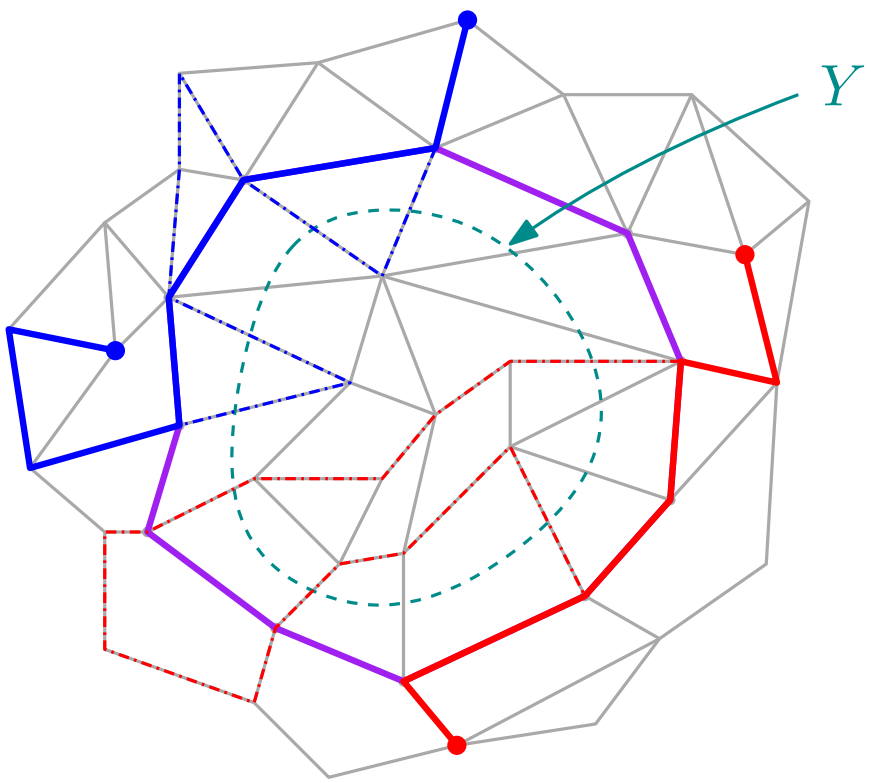
If $G - Y$ is a yes-instance of 2DPP,
then G is a yes-instance of 2DPP

because the 2 disjoint paths of $G - Y$ are also disjoint paths of G.

If G is a yes-instance of 2DPP,
then $G - Y$ is a yes-instance of 2DPP

because for any 2 disjoint paths of G, there exist alternative disjoint paths of G that do not intersect Y.

Easy case: 2DPP on planar graphs



If $G - Y$ is a yes-instance of 2DPP,
then G is a yes-instance of 2DPP

because the 2 disjoint paths of $G - Y$ are also disjoint paths of G.

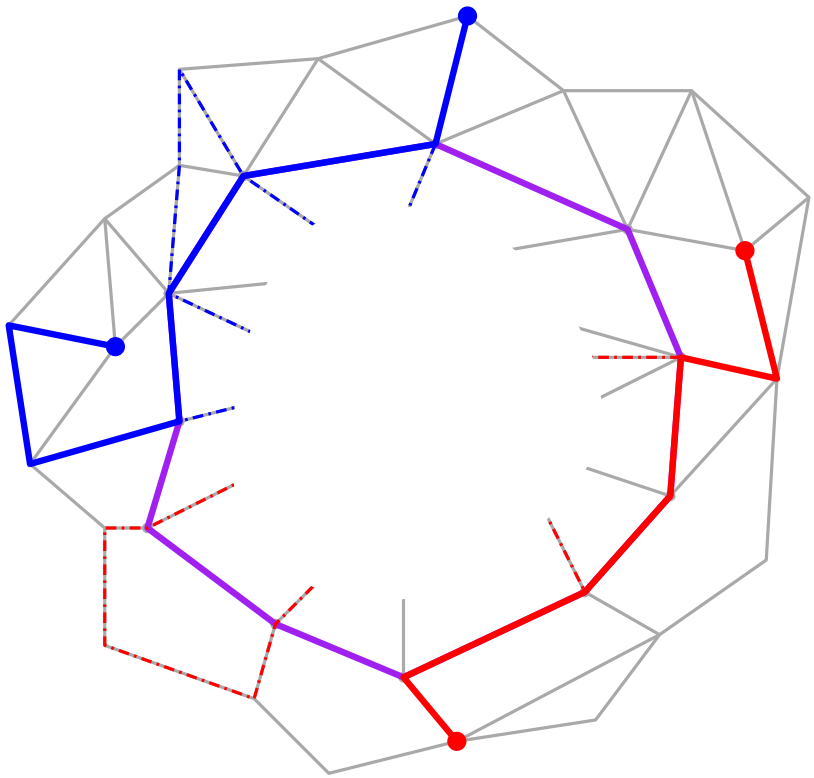
If G is a yes-instance of 2DPP,
then $G - Y$ is a yes-instance of 2DPP

because for any 2 disjoint paths of G, there exist alternative disjoint paths of G that do not intersect Y.

$\Longrightarrow$ G and $G - Y$ are equivalent instances of 2DPP.

The vertices of Y are **irrelevant**.

Easy case: 2DPP on planar graphs

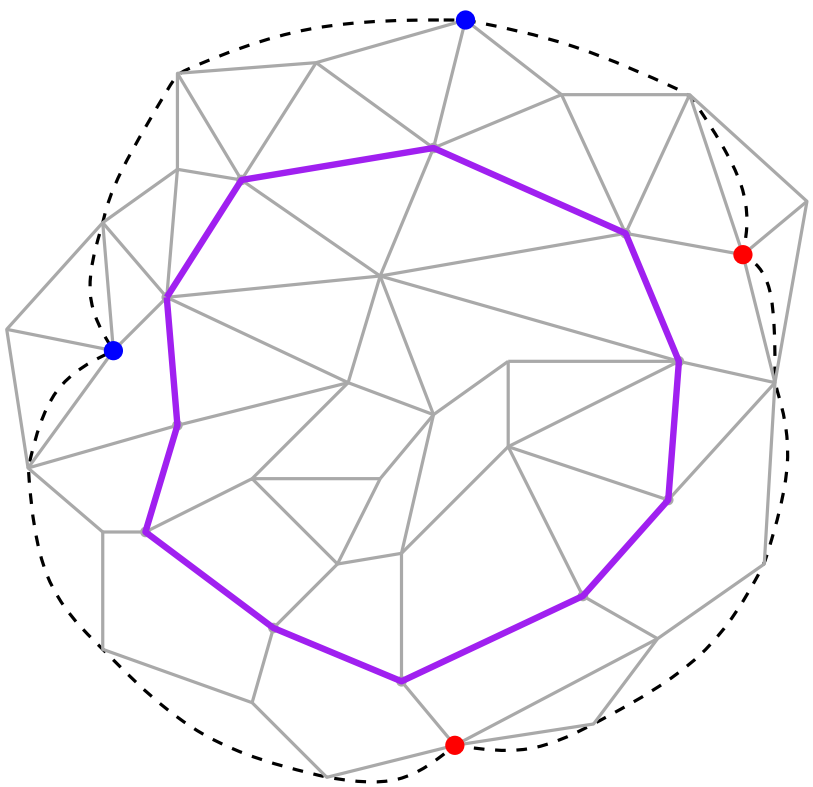


$\Longrightarrow$ can delete Y and recurse.

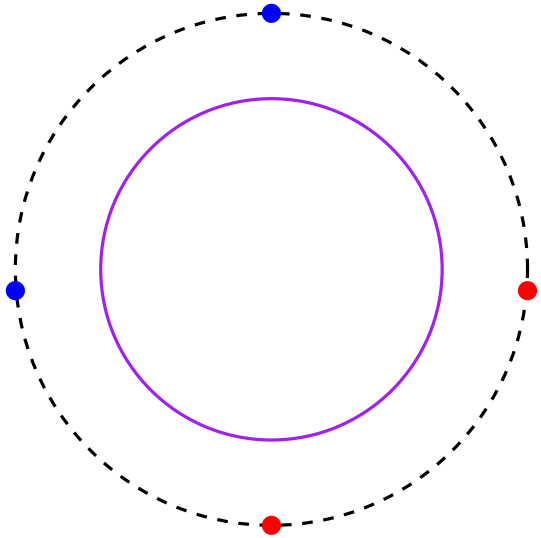
$\Longrightarrow$ G and $G - Y$ are equivalent instances of 2DPP.

The vertices of Y are **irrelevant**.

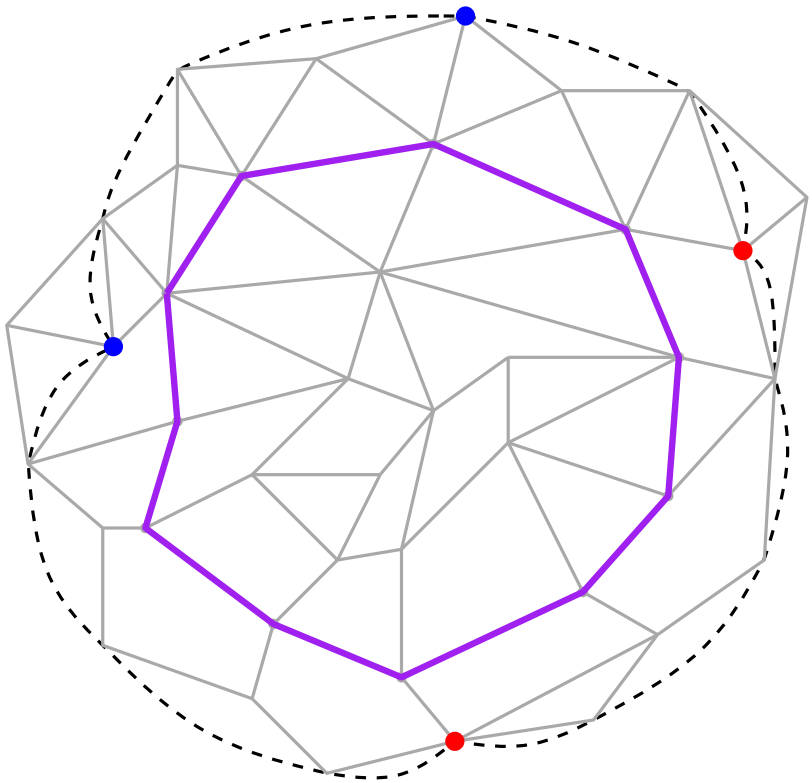
Easy case: 2DPP on planar graphs



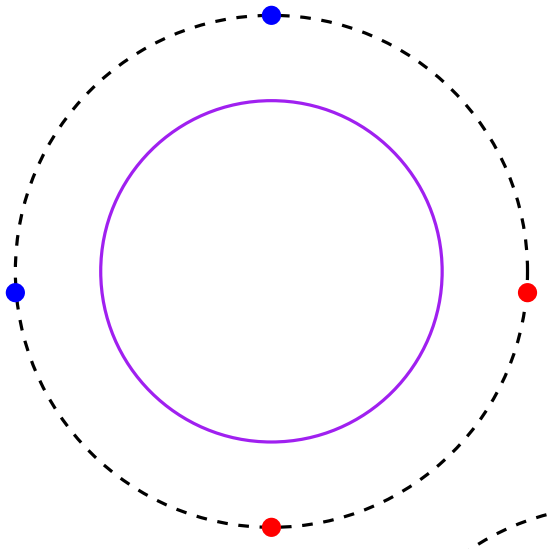
Simplified representation:

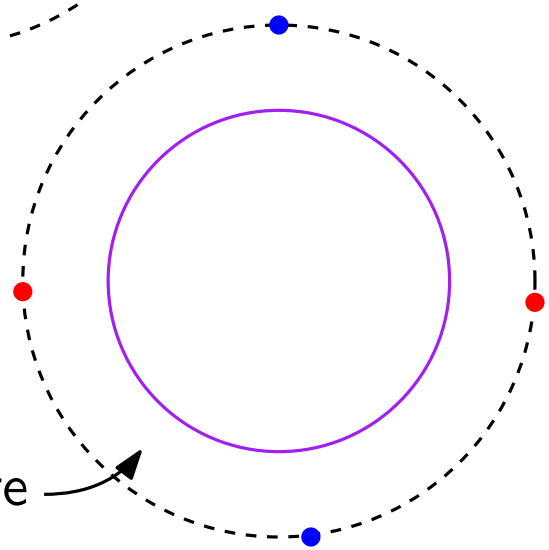


Easy case: 2DPP on planar graphs



Simplified representation:

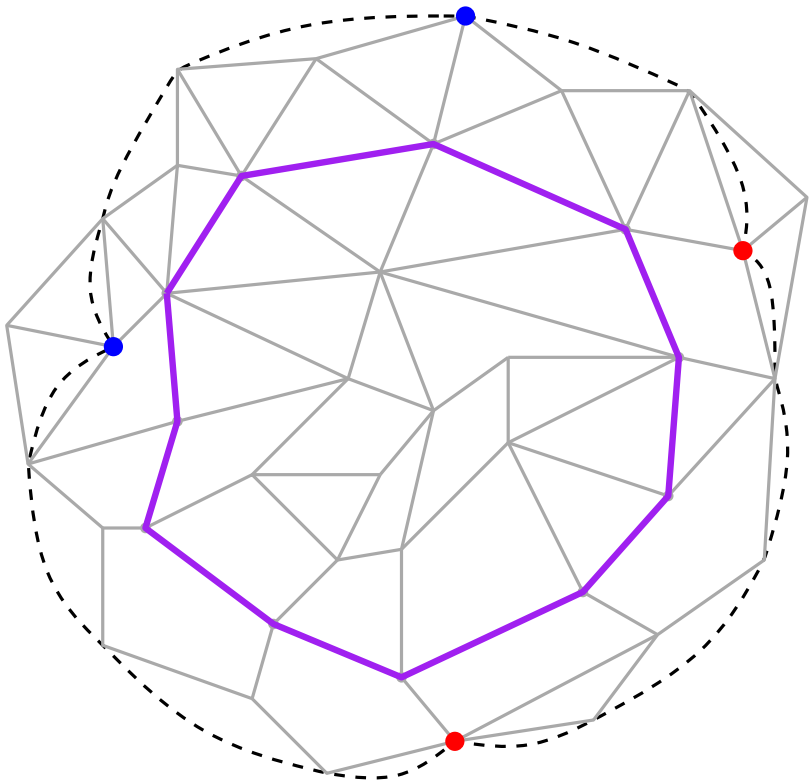




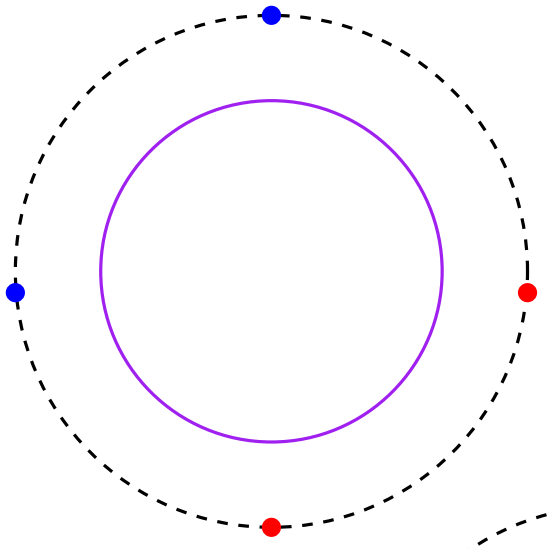
The order on the boundary matters!

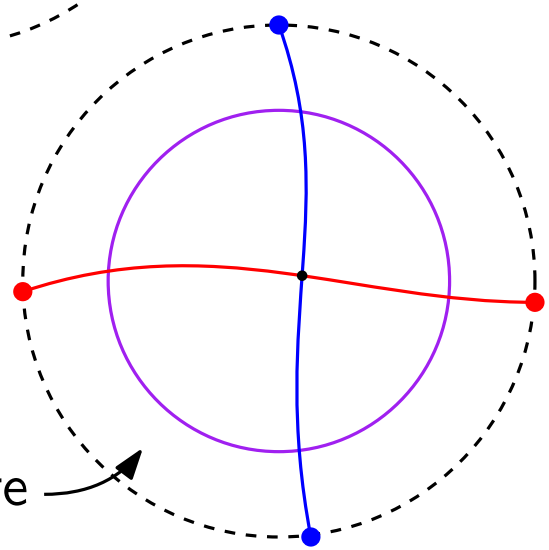
no disjoint paths here

Easy case: 2DPP on planar graphs



Simplified representation:

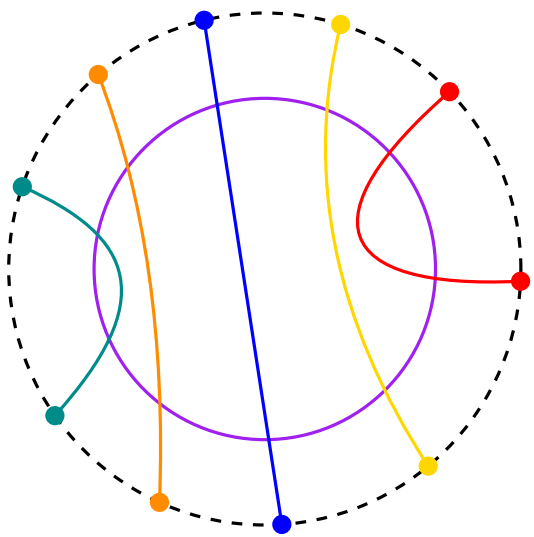




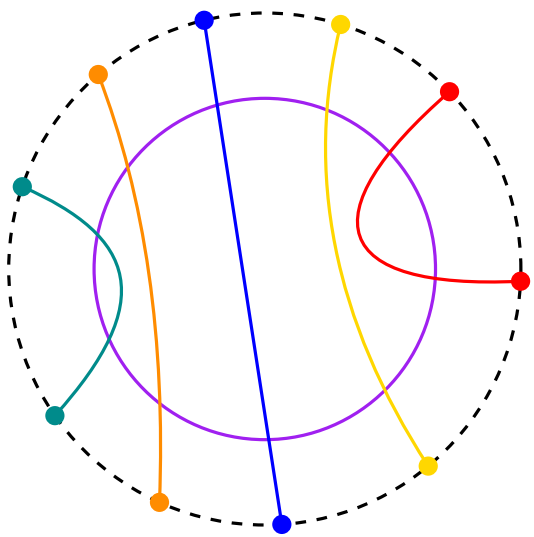
The order on the boundary matters!

no disjoint paths here

Slightly more difficult case: kDPP on planar graphs

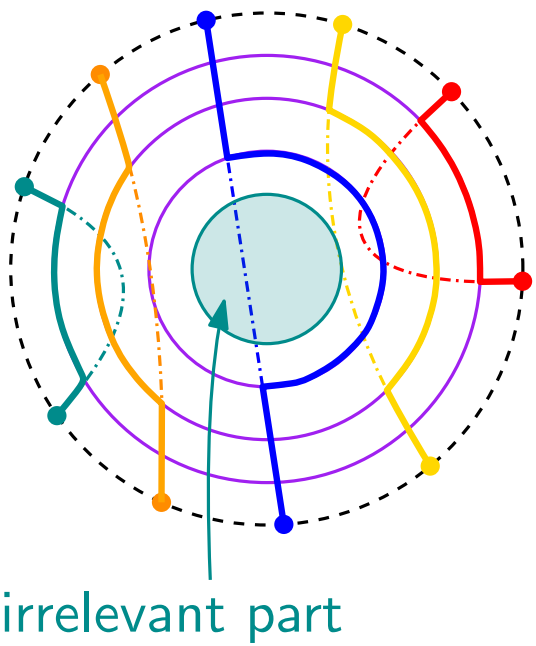


Slightly more difficult case: kDPP on planar graphs



One cycle is not enough to reroute paths.

Slightly more difficult case: kDPP on planar graphs



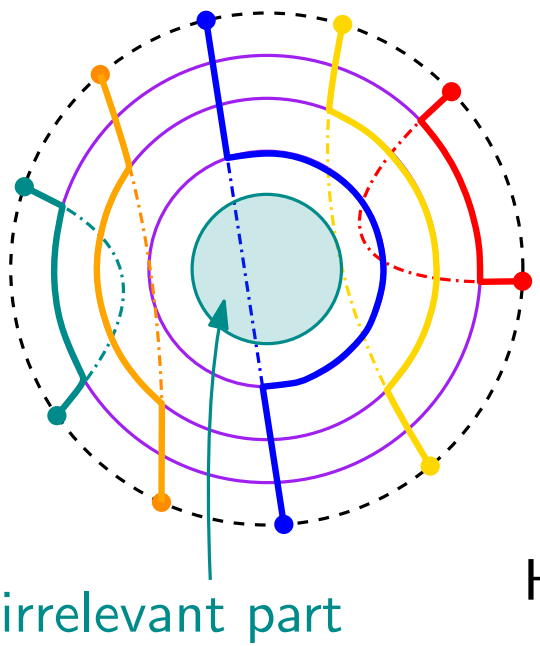
One cycle is not enough to reroute paths.

[Graph Minors XXI]

Unique Linkage theorem (Corollary)

If G constains $f(k)$ nested cycles, with all terminals outside the outer cycle, then all paths of a solution of kDPP can be rerouted away from the inner cycle.

Slightly more difficult case: kDPP on planar graphs



One cycle is not enough to reroute paths.

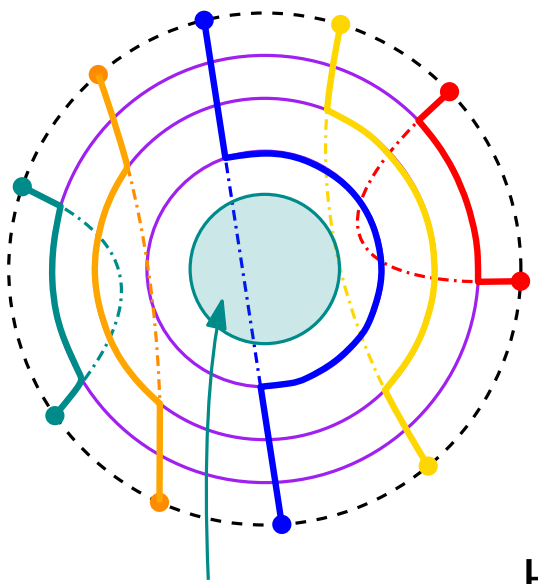
[Graph Minors XXI]

Unique Linkage theorem (Corollary)

If G constains $f(k)$ nested cycles, with all terminals outside the outer cycle, then all paths of a solution of kDPP can be rerouted away from the inner cycle.

How to solve kDPP?

Slightly more difficult case: kDPP on planar graphs



irrelevant part

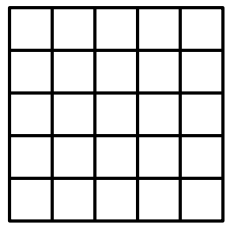
One cycle is not enough to reroute paths.

[Graph Minors XXI]

Unique Linkage theorem (Corollary)

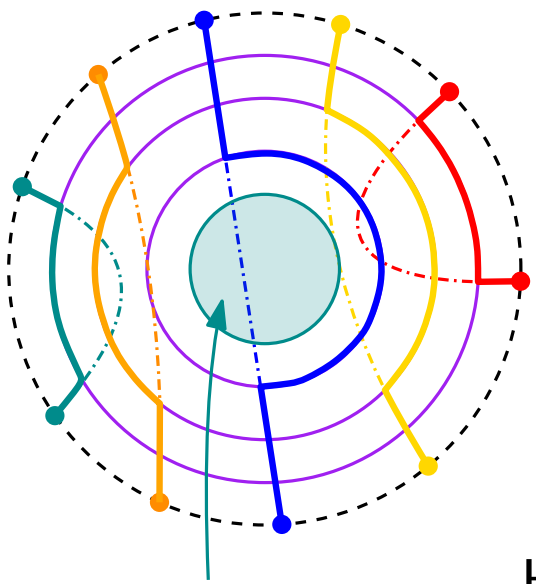
If G constains $f(k)$ nested cycles, with all terminals outside the outer cycle, then all paths of a solution of kDPP can be rerouted away from the inner cycle.

How to solve kDPP? Win/Win:



If G has treewidth $\geq f_1(k)$: (Grid Exclusion theorem)
it contains a grid of height $f_2(k)$ as a minor

Slightly more difficult case: kDPP on planar graphs



irrelevant part

One cycle is not enough to reroute paths.

[Graph Minors XXI]

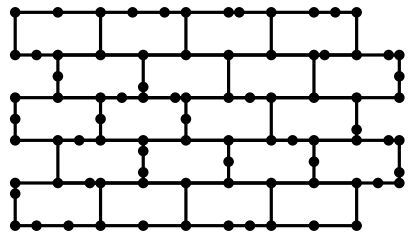
Unique Linkage theorem (Corollary)

If G constains $f(k)$ nested cycles, with all terminals outside the outer cycle, then all paths of a solution of kDPP can be rerouted away from the inner cycle.

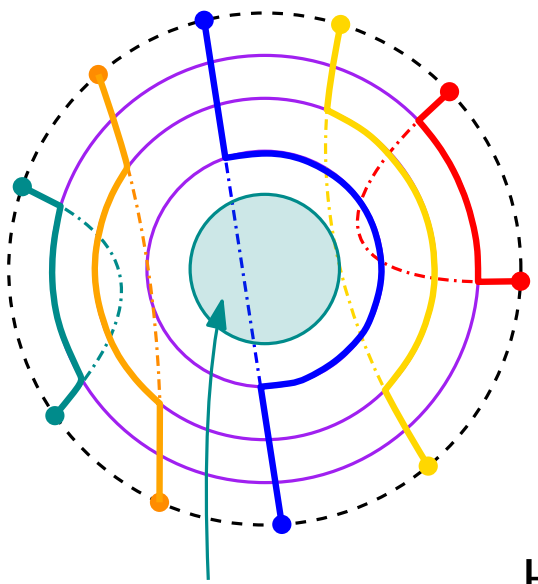
How to solve kDPP? Win/Win:

If G has treewidth $\geq f_1(k)$:
it contains a grid of height $f_2(k)$ as a minor
$\Leftrightarrow$ it contains a wall of height $f_2(k)$ as a subgraph

easier to find



Slightly more difficult case: kDPP on planar graphs



irrelevant part

One cycle is not enough to reroute paths.

[Graph Minors XXI]

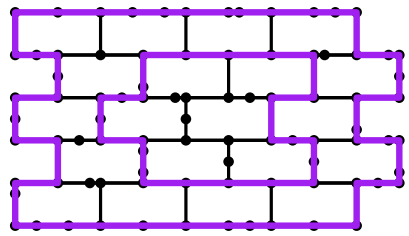
Unique Linkage theorem (Corollary)

If G constains $f(k)$ nested cycles, with all terminals outside the outer cycle, then all paths of a solution of kDPP can be rerouted away from the inner cycle.

How to solve kDPP? Win/Win:

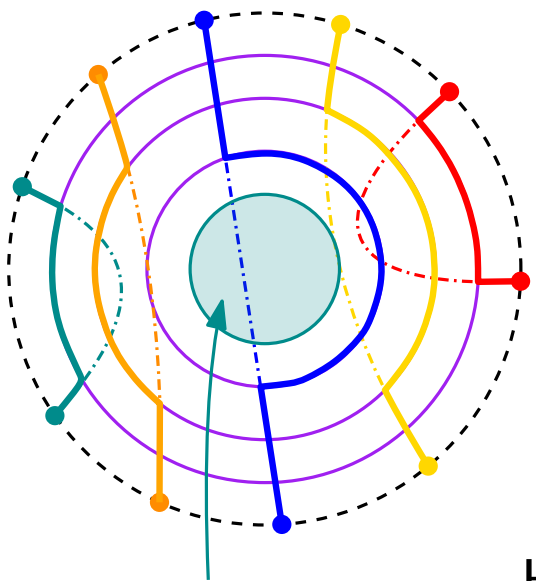
If G has treewidth $\geq f_1(k)$:
it contains a grid of height $f_2(k)$ as a minor
$\Leftrightarrow$ it contains a wall of height $f_2(k)$ as a subgraph

easier to find



nested cycles

Slightly more difficult case: kDPP on planar graphs



irrelevant part

One cycle is not enough to reroute paths.

[Graph Minors XXI]

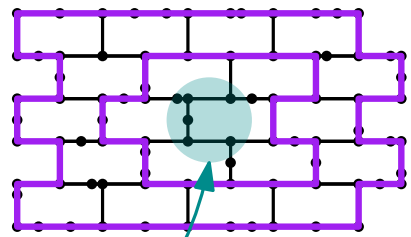
Unique Linkage theorem (Corollary)

If G constains $f(k)$ nested cycles, with all terminals outside the outer cycle, then all paths of a solution of kDPP can be rerouted away from the inner cycle.

How to solve kDPP? Win/Win:

If G has treewidth $\geq f_1(k)$:
it contains a grid of height $f_2(k)$ as a minor
$\Leftrightarrow$ it contains a wall of height $f_2(k)$ as a subgraph

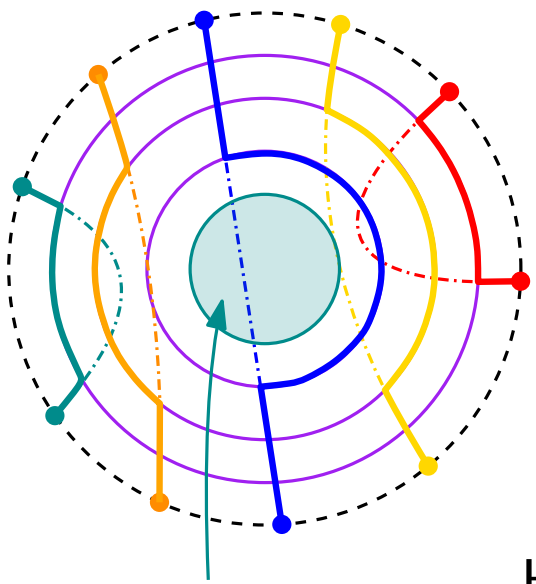
easier to find



nested cycles

center of the wall is irrelevant:
delete and recurse

Slightly more difficult case: kDPP on planar graphs



irrelevant part

One cycle is not enough to reroute paths.

[Graph Minors XXI]

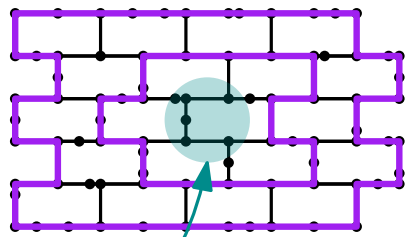
Unique Linkage theorem (Corollary)

If G constains $f(k)$ nested cycles, with all terminals outside the outer cycle, then all paths of a solution of kDPP can be rerouted away from the inner cycle.

How to solve kDPP? Win/Win:

If G has treewidth $\geq f_1(k)$:
it contains a grid of height $f_2(k)$ as a minor
$\Leftrightarrow$ it contains a wall of height $f_2(k)$ as a subgraph

easier to find



If G has treewidth $< f_1(k)$:
conclude with Courcelle's theorem

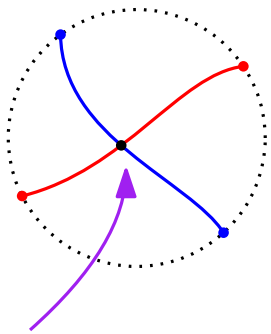
nested cycles

center of the wall is irrelevant:
delete and recurse

Far more difficult case: kDPP on general graphs

Far more difficult case: kDPP on general graphs

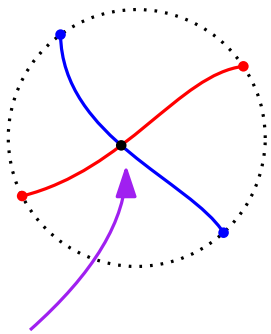
Planar case:



any 2 paths that cross one another must intersect

Far more difficult case: kDPP on general graphs

Planar case:

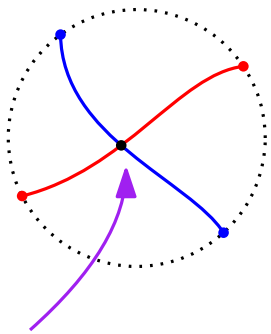


any 2 paths that cross one another must intersect

How to get a similar property on general graphs?

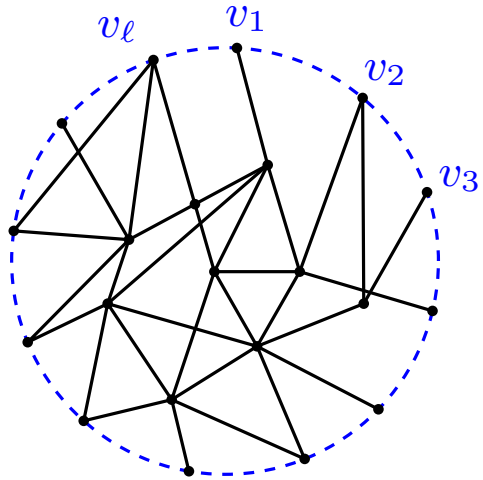
Far more difficult case: kDPP on general graphs

Planar case:



any 2 paths that cross one another must intersect

How to get a similar property on general graphs?

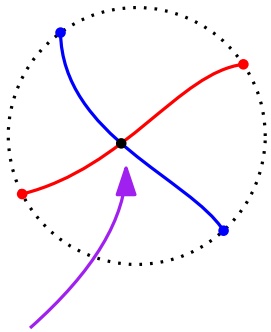


Society: pair (G, Ω)

cyclic ordering with $V(\Omega) = \{v_1, v_2, \ldots, v_\ell\}$.

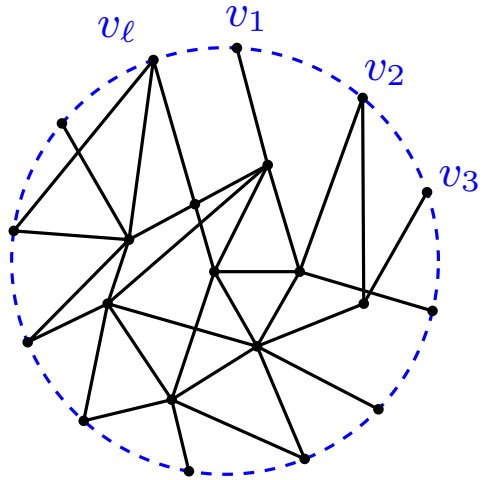
Far more difficult case: kDPP on general graphs

Planar case:



any 2 paths that cross one another must intersect

How to get a similar property on general graphs?

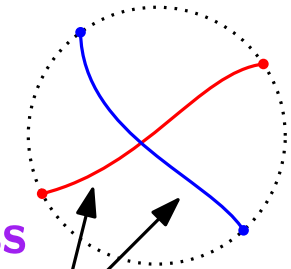


Society: pair (G, Ω)

cyclic ordering with $V(\Omega) = \{v_1, v_2, \ldots, v_\ell\}$.

[Graph minor XIII]

A society has no cross if and only if it admits a **rural rendition**.

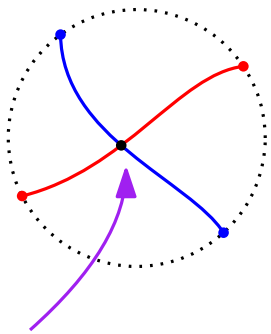


a **cross**

disjoint paths

Far more difficult case: kDPP on general graphs

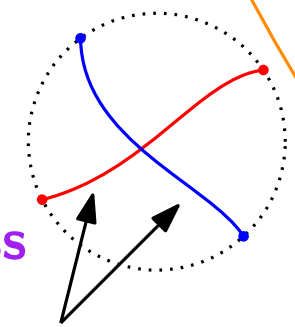
Planar case:



any 2 paths that cross one another must intersect

[Graph minor XIII]

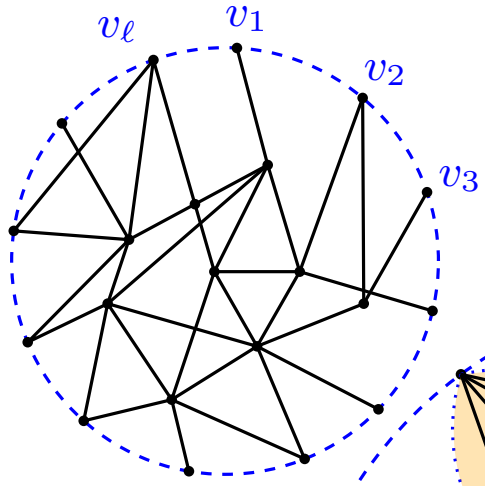
A society has no cross if and only if it admits a **rural rendition**.



a **cross**

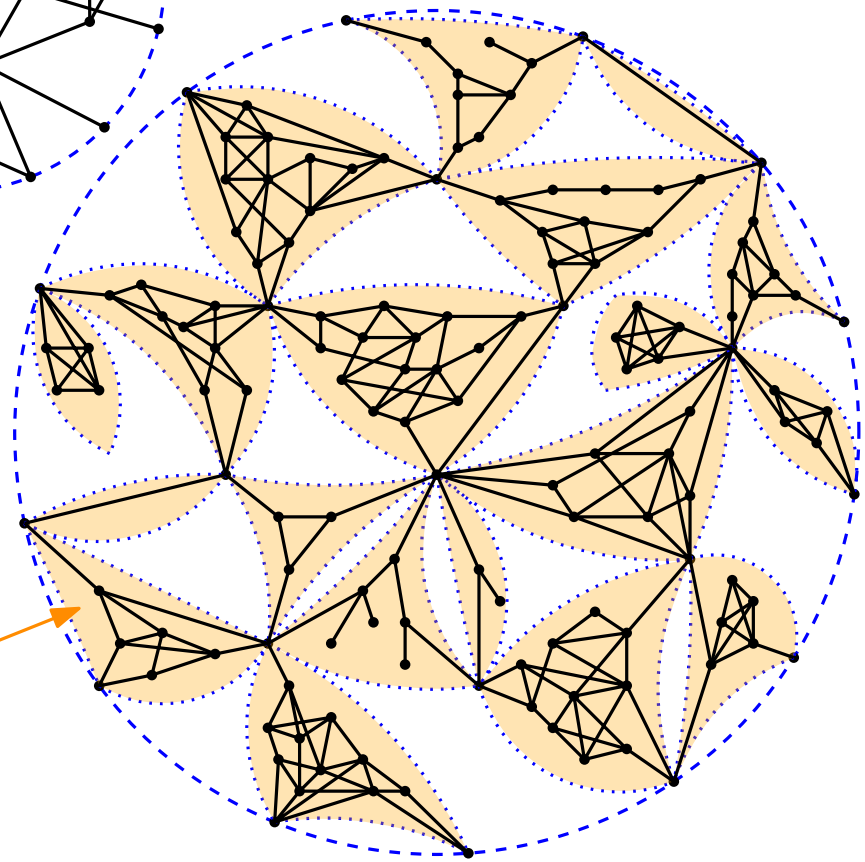
disjoint paths

How to get a similar property on general graphs?



Society: pair (G, Ω)

cyclic ordering with $V(\Omega) = \{v_1, v_2, \ldots, v_\ell\}$.

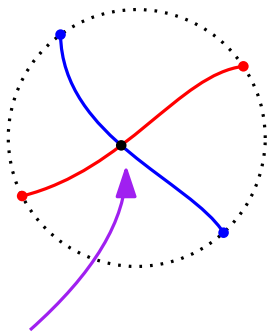


cell

boundary cell ≤ 3

Far more difficult case: kDPP on general graphs

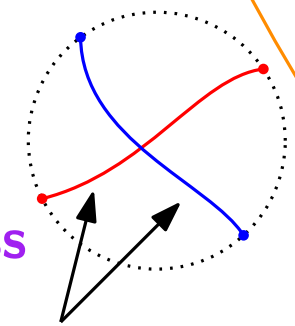
Planar case:



any 2 paths that cross one another must intersect

[Graph minor XIII]

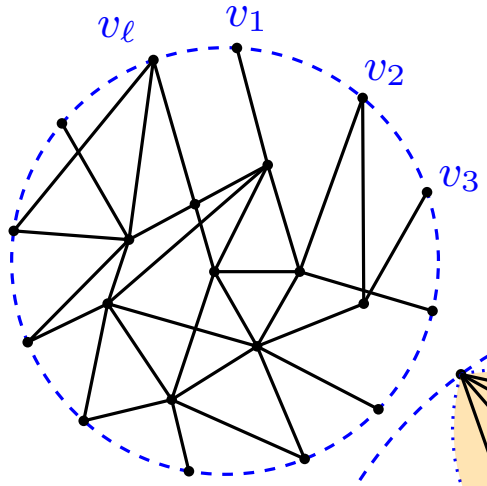
A society has no cross if and only if it admits a **rural rendition**.



a **cross**

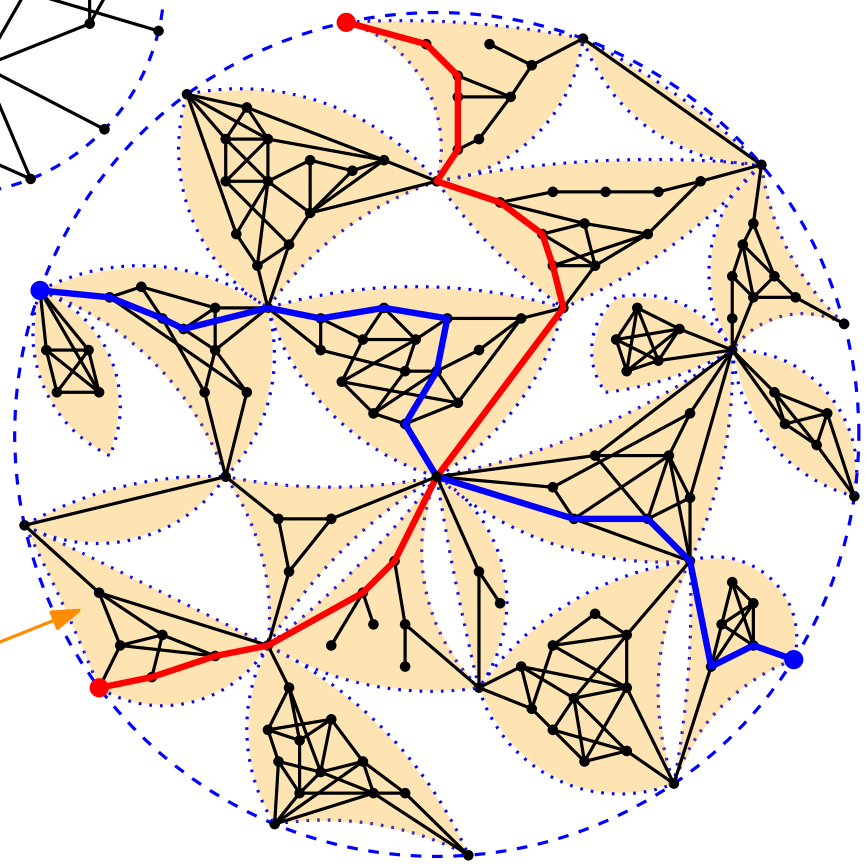
disjoint paths

How to get a similar property on general graphs?



Society: pair (G, Ω)

cyclic ordering with $V(\Omega) = \{v_1, v_2, \ldots, v_\ell\}$.



cell

boundary cell ≤ 3

Far more difficult case: kDPP on general graphs

[Graph Minors XIII]

Flat Wall theorem

FPT algo in k and t

A graph G either has:

- treewidth at most $f(k,t)$, or
- a clique of size t as a minor, or
- a vertex set $A \leq g(t)$ and a wall W of height k s.t. W is a **flat wall** of $G - A$.

Far more difficult case: kDPP on general graphs

[Graph Minors XIII]

Flat Wall theorem

FPT algo in k and t

A graph G either has:

- treewidth at most $f(k,t)$, or
- a clique of size t as a minor, or
- a vertex set $A \leq g(t)$ and a wall W of height k s.t. W is a **flat wall** of $G - A$.

conclude with Courcelle's theorem

Far more difficult case: kDPP on general graphs

[Graph Minors XIII]

Flat Wall theorem

FPT algo in k and t

A graph G either has:

- treewidth at most $f(k,t)$, or
- a clique of size t as a minor, or
- a vertex set $A \leq g(t)$ and a wall W of height k s.t. W is a **flat wall** of $G - A$.

not covered here (can also find some kind of irrelevant vertex)

Far more difficult case: kDPP on general graphs

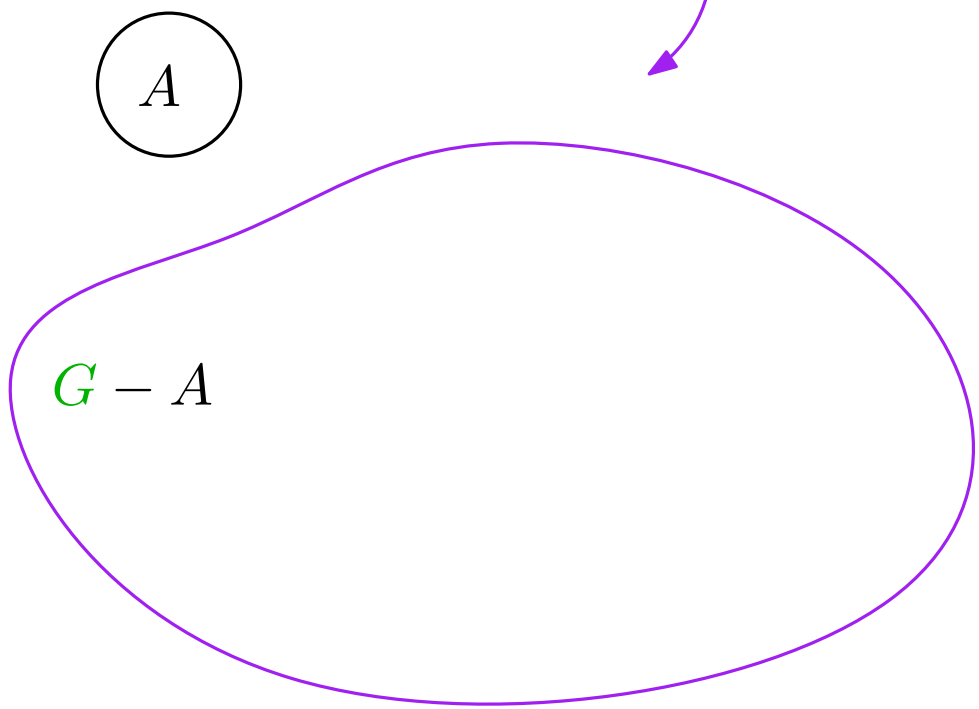
[Graph Minors XIII]

Flat Wall theorem

FPT algo in k and t

A graph G either has:

- treewidth at most $f(k,t)$, or
- a clique of size t as a minor, or
- a vertex set $A \leq g(t)$ and a wall W of height k s.t. W is a **flat wall** of $G - A$.



Far more difficult case: kDPP on general graphs

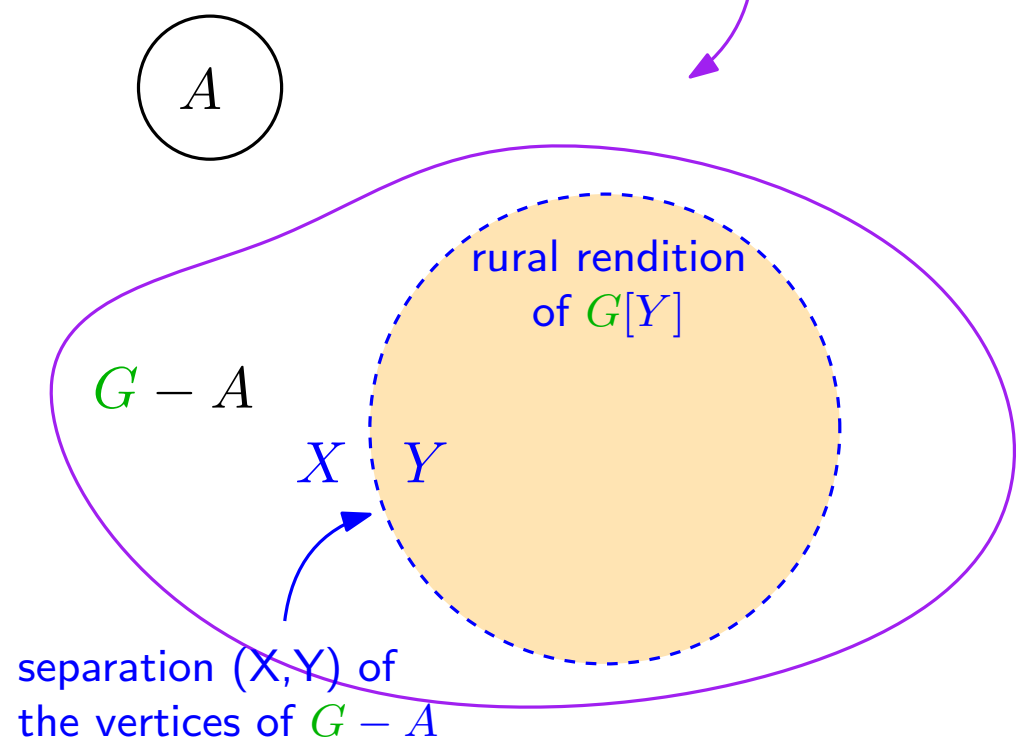
[Graph Minors XIII]

Flat Wall theorem

FPT algo in k and t

A graph G either has:

- treewidth at most $f(k,t)$, or
- a clique of size t as a minor, or
- a vertex set $A \leq g(t)$ and a wall W of height k s.t. W is a **flat wall** of $G - A$.



Far more difficult case: kDPP on general graphs

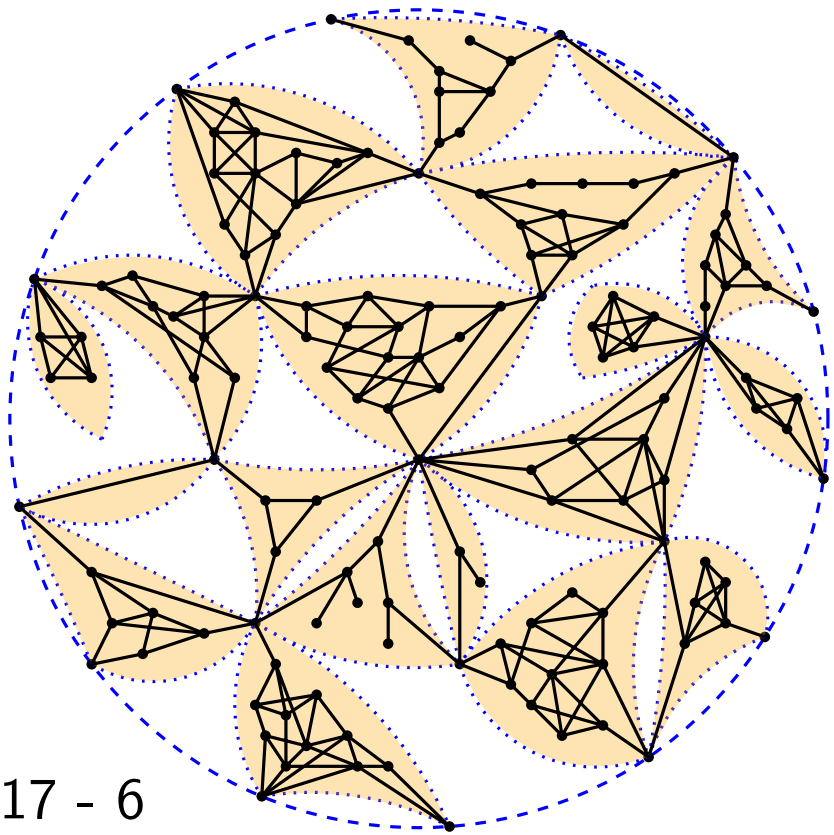
[Graph Minors XIII]

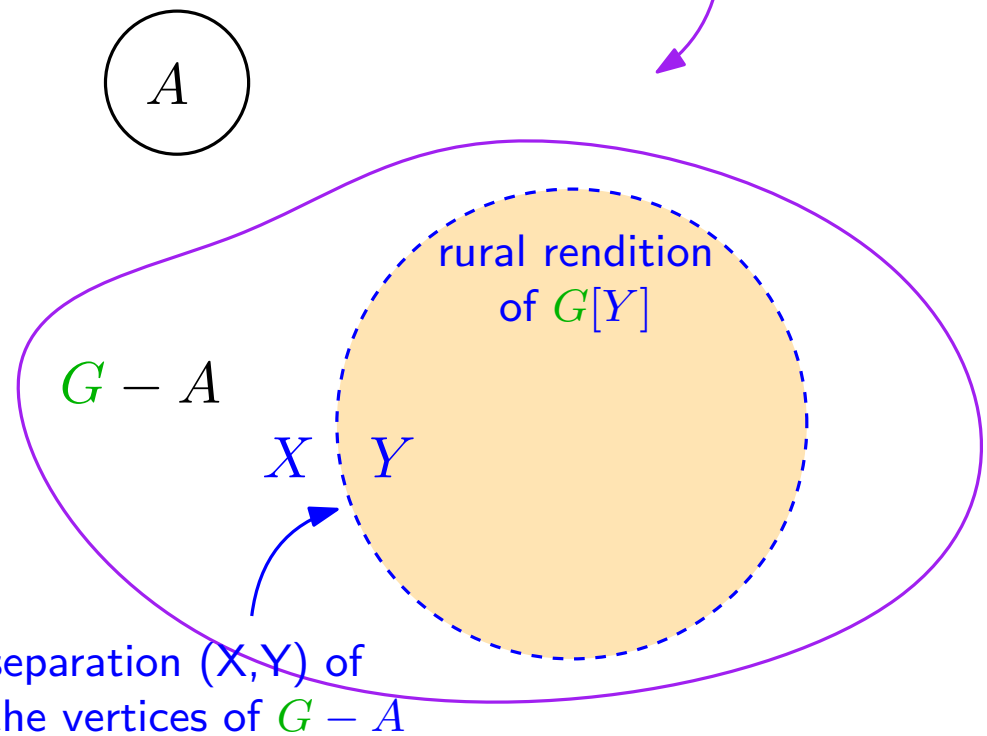
Flat Wall theorem

FPT algo in k and t

A graph G either has:

- treewidth at most $f(k,t)$, or
- a clique of size t as a minor, or
- a vertex set $A \leq g(t)$ and a wall W of height k s.t. W is a **flat wall** of $G - A$.





Far more difficult case: kDPP on general graphs

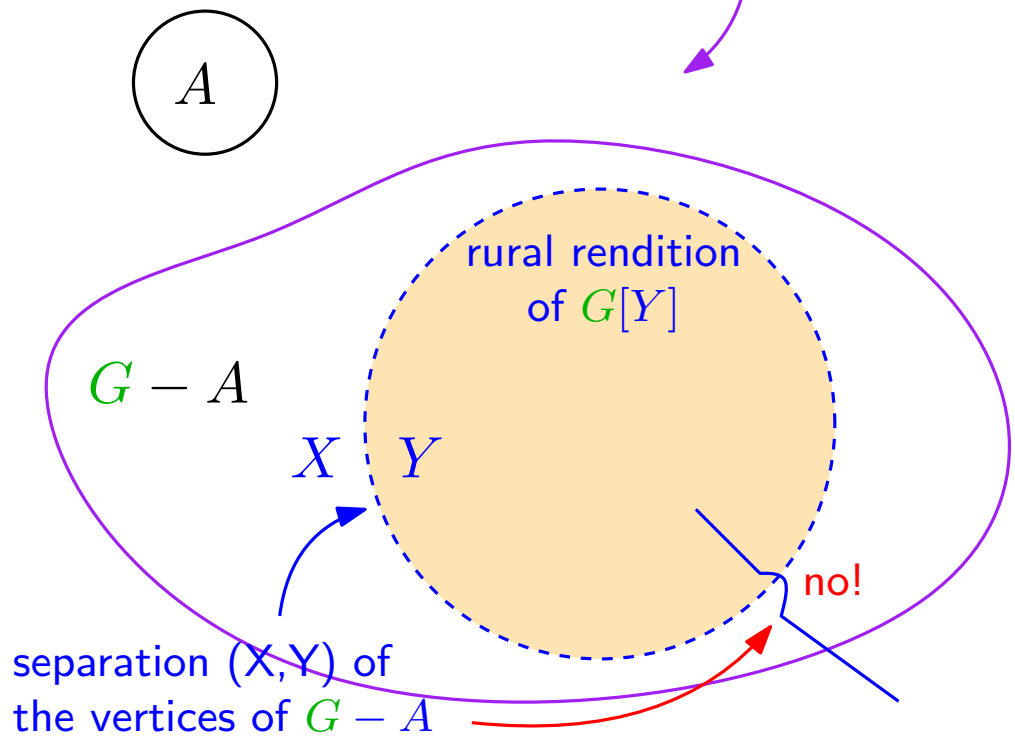
[Graph Minors XIII]

Flat Wall theorem

FPT algo in k and t

A graph G either has:

- treewidth at most $f(k,t)$, or
- a clique of size t as a minor, or
- a vertex set $A \leq g(t)$ and a wall W of height k s.t. W is a **flat wall** of $G - A$.



Far more difficult case: kDPP on general graphs

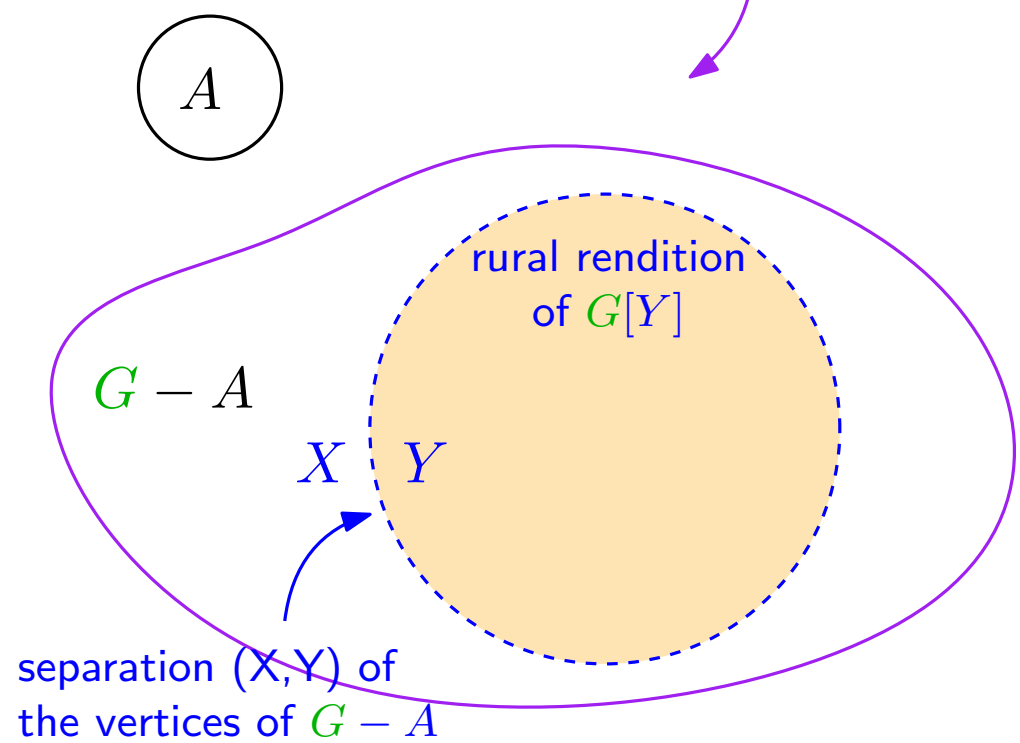
[Graph Minors XIII]

Flat Wall theorem

FPT algo in k and t

A graph G either has:

- treewidth at most $f(k,t)$, or
- a clique of size t as a minor, or
- a vertex set $A \leq g(t)$ and a wall W of height k s.t. W is a **flat wall** of $G - A$.



Far more difficult case: kDPP on general graphs

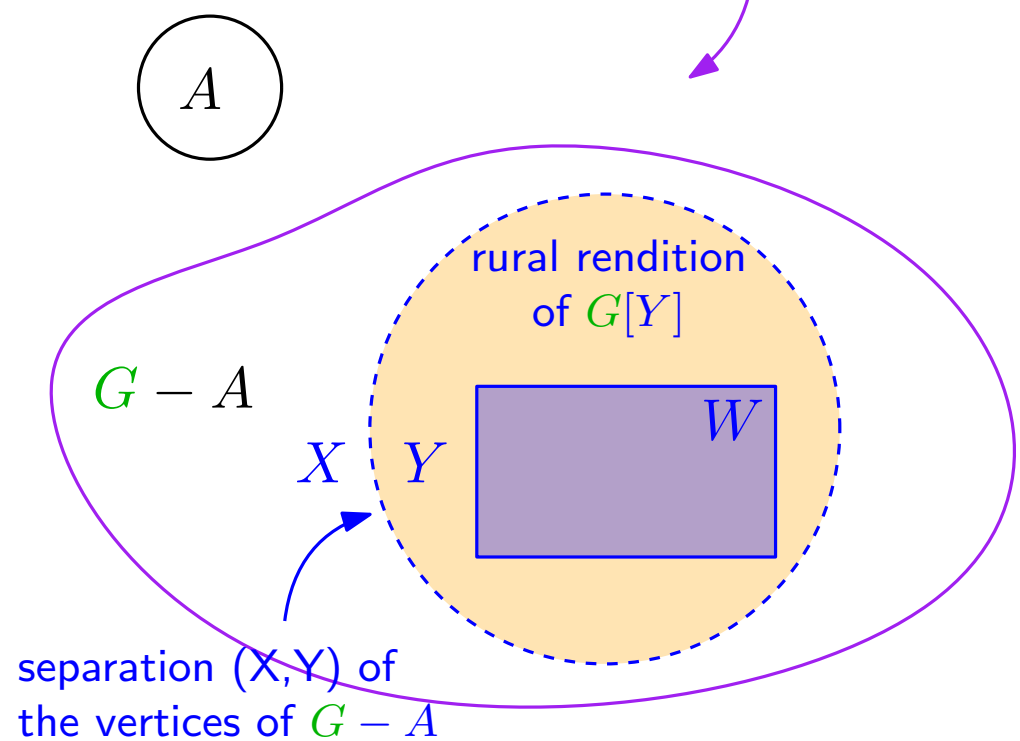
[Graph Minors XIII]

Flat Wall theorem

FPT algo in k and t

A graph G either has:

- treewidth at most $f(k,t)$, or
- a clique of size t as a minor, or
- a vertex set $A \leq g(t)$ and a wall W of height k s.t. W is a **flat wall** of $G - A$.



Far more difficult case: kDPP on general graphs

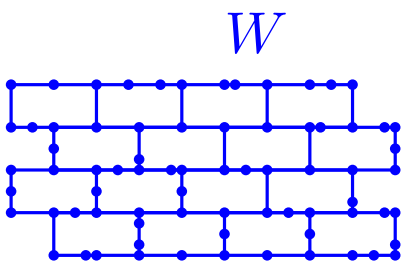
[Graph Minors XIII]

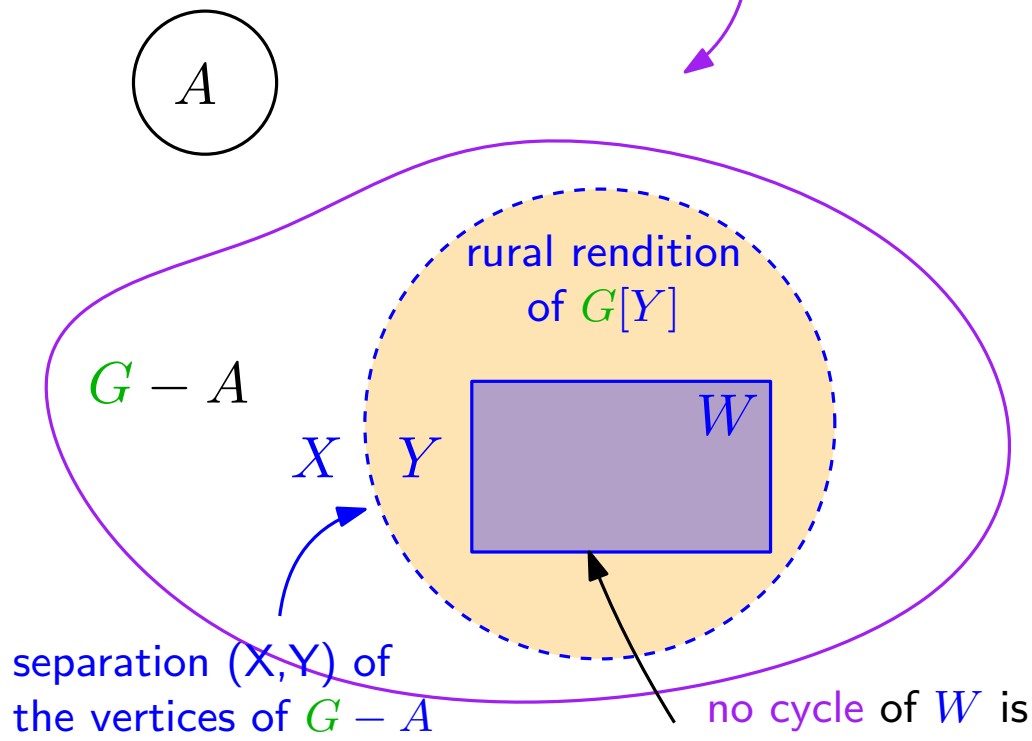
Flat Wall theorem

FPT algo in k and t

A graph G either has:

- treewidth at most $f(k,t)$, or
- a clique of size t as a minor, or
- a vertex set $A \leq g(t)$ and a wall W of height k s.t. W is a **flat wall** of $G - A$.





Far more difficult case: kDPP on general graphs

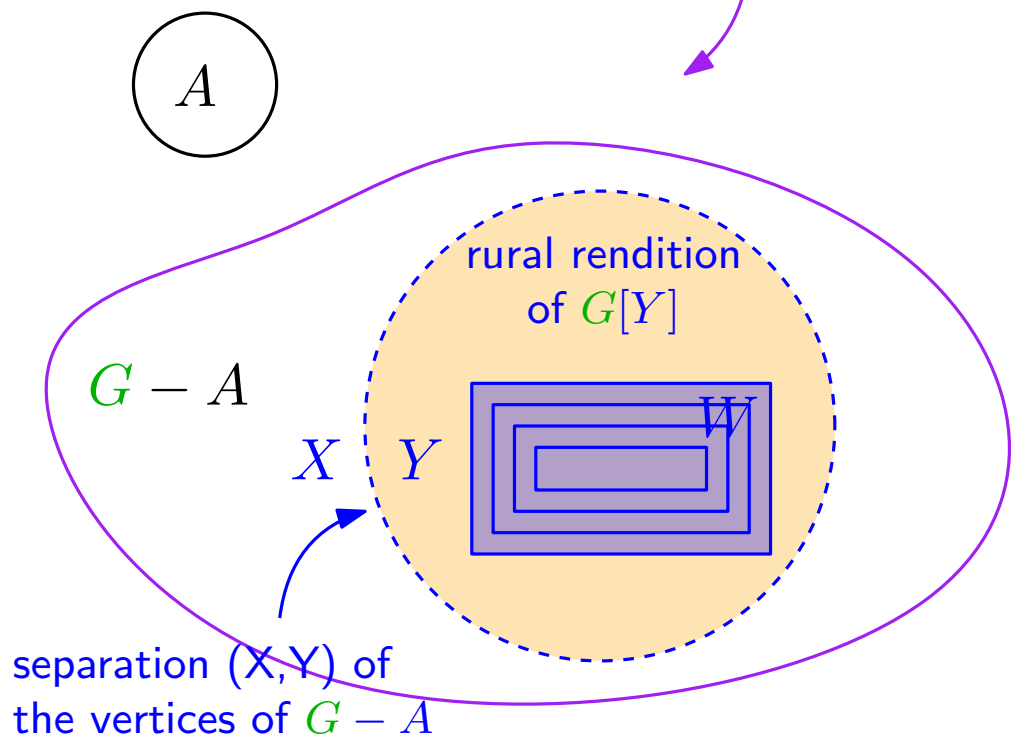
[Graph Minors XIII]

Flat Wall theorem

FPT algo in k and t

A graph G either has:

- treewidth at most $f(k,t)$, or
- a clique of size t as a minor, or
- a vertex set $A \leq g(t)$ and a wall W of height k s.t. W is a **flat wall** of $G - A$.



Far more difficult case: kDPP on general graphs

[Graph Minors XIII]

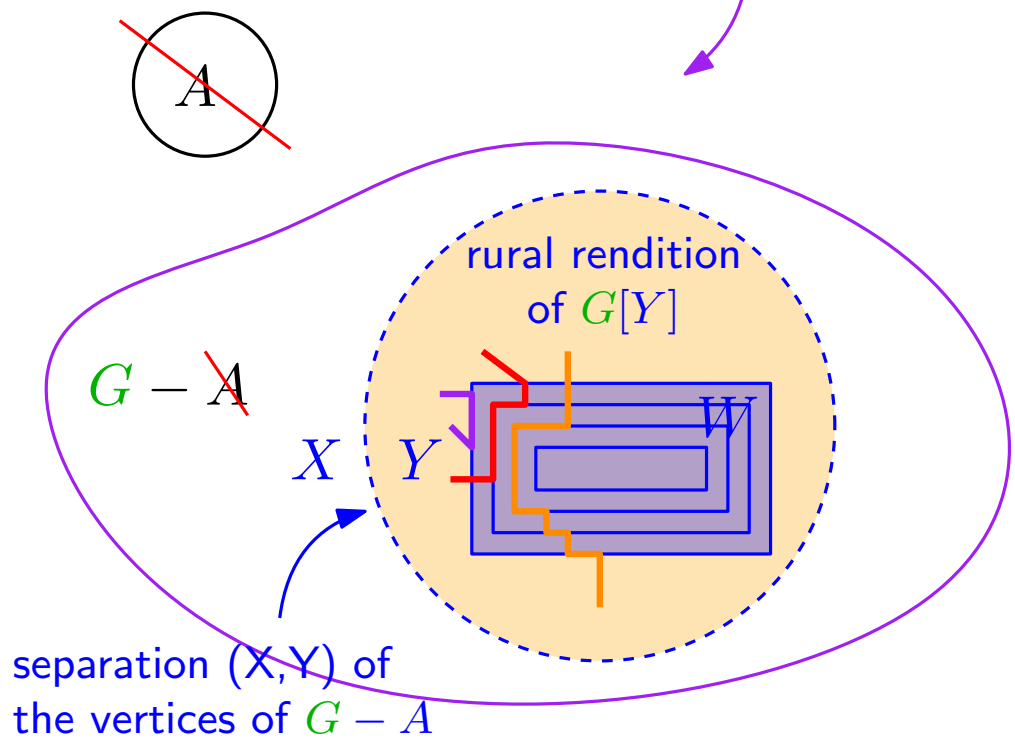
Flat Wall theorem

FPT algo in k and t

A graph G either has:

- treewidth at most $f(k,t)$, or
- a clique of size t as a minor, or
- a vertex set $A \leq g(t)$ and a wall W of height k s.t. W is a **flat wall** of $G - A$.

If $A = \emptyset$, as before:
Unique Linkage theorem



Far more difficult case: kDPP on general graphs

[Graph Minors XIII]

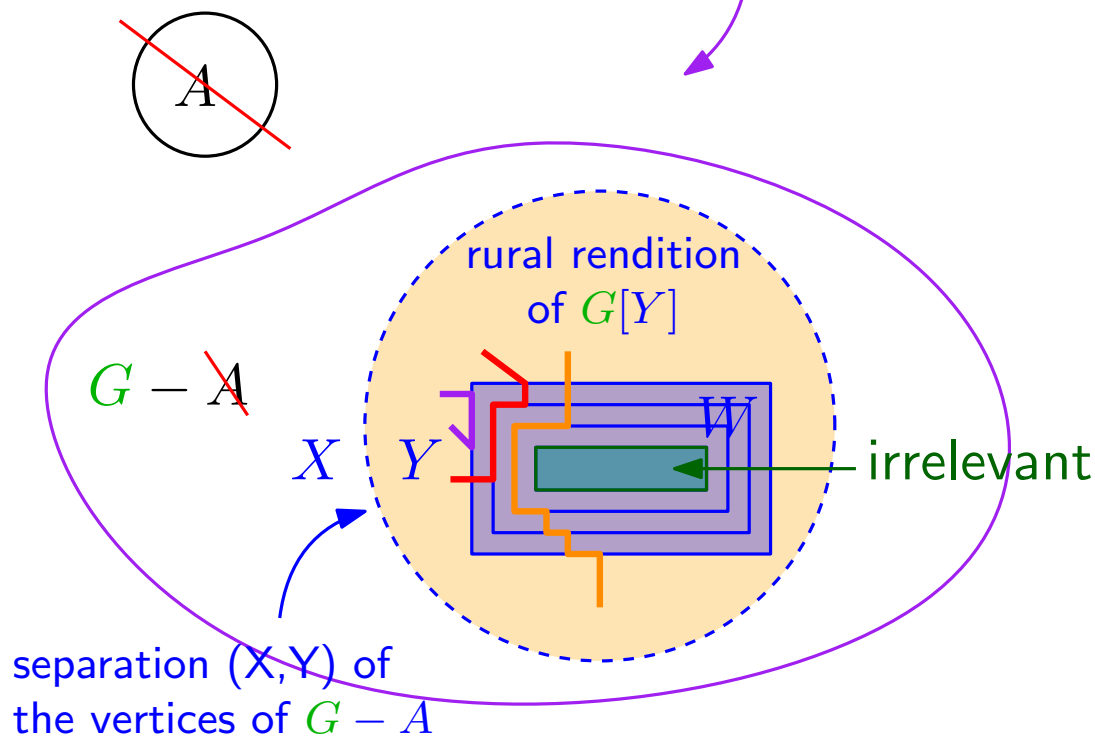
Flat Wall theorem

FPT algo in k and t

A graph G either has:

- treewidth at most $f(k,t)$, or
- a clique of size t as a minor, or
- a vertex set $A \leq g(t)$ and a wall W of height k s.t. W is a **flat wall** of $G - A$.

If $A = \emptyset$, as before:
Unique Linkage theorem



Far more difficult case: kDPP on general graphs

[Graph Minors XIII]

Flat Wall theorem

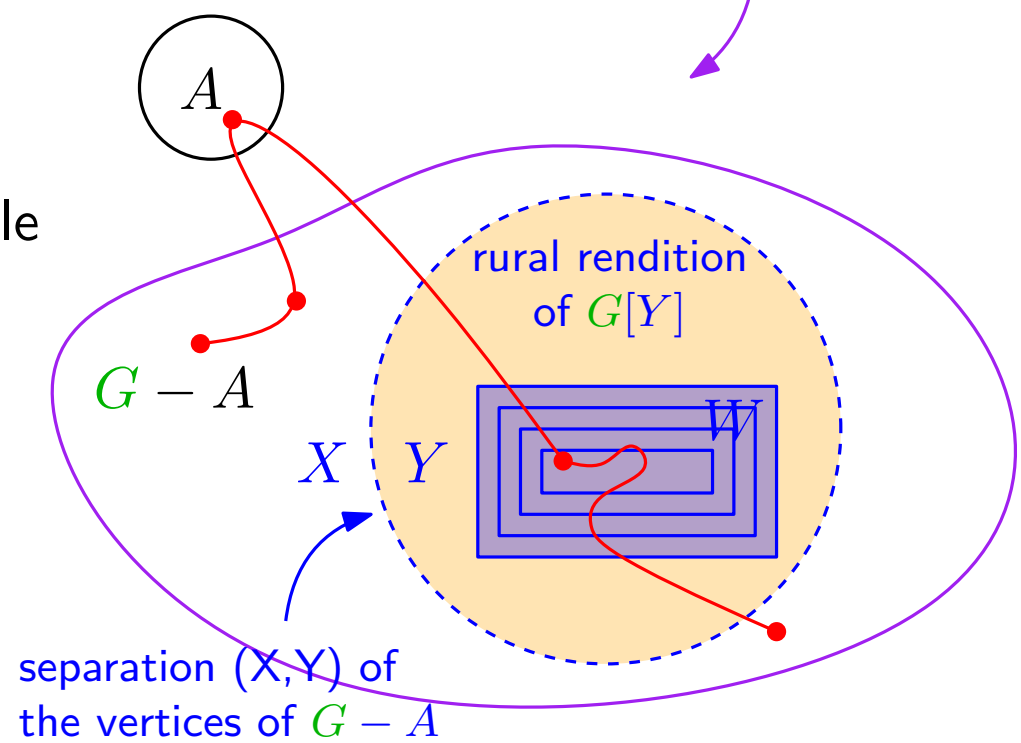
FPT algo in k and t

A graph G either has:

- treewidth at most $f(k,t)$, or
- a clique of size t as a minor, or
- a vertex set $A \leq g(t)$ and a wall W of height k s.t. W is a **flat wall** of $G - A$.

Problem if $A \neq \emptyset$:

a path might go through A in W:
cannot always reroute through a cycle



Far more difficult case: kDPP on general graphs

[Graph Minors XIII]

Flat Wall theorem

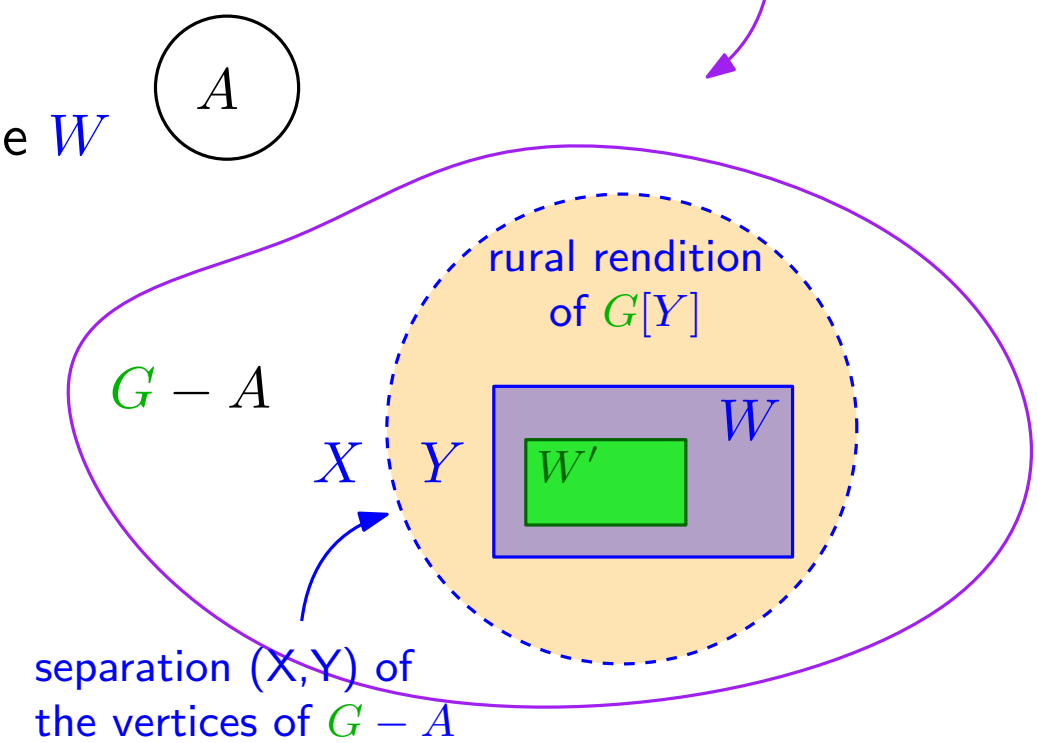
FPT algo in k and t

A graph G either has:

- treewidth at most $f(k,t)$, or
- a clique of size t as a minor, or
- a vertex set $A \leq g(t)$ and a wall W of height k s.t. W is a **flat wall** of $G - A$.

Solution:

find a **homogeneous wall** W' inside W



Far more difficult case: kDPP on general graphs

[Graph Minors XIII]

Flat Wall theorem

FPT algo in k and t

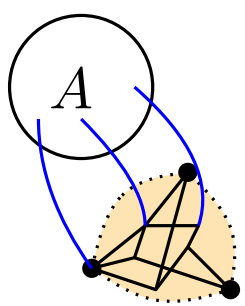
A graph G either has:

- treewidth at most $f(k,t)$, or
- a clique of size t as a minor, or
- a vertex set $A \leq g(t)$ and a wall W of height k s.t. W is a **flat wall** of $G - A$.

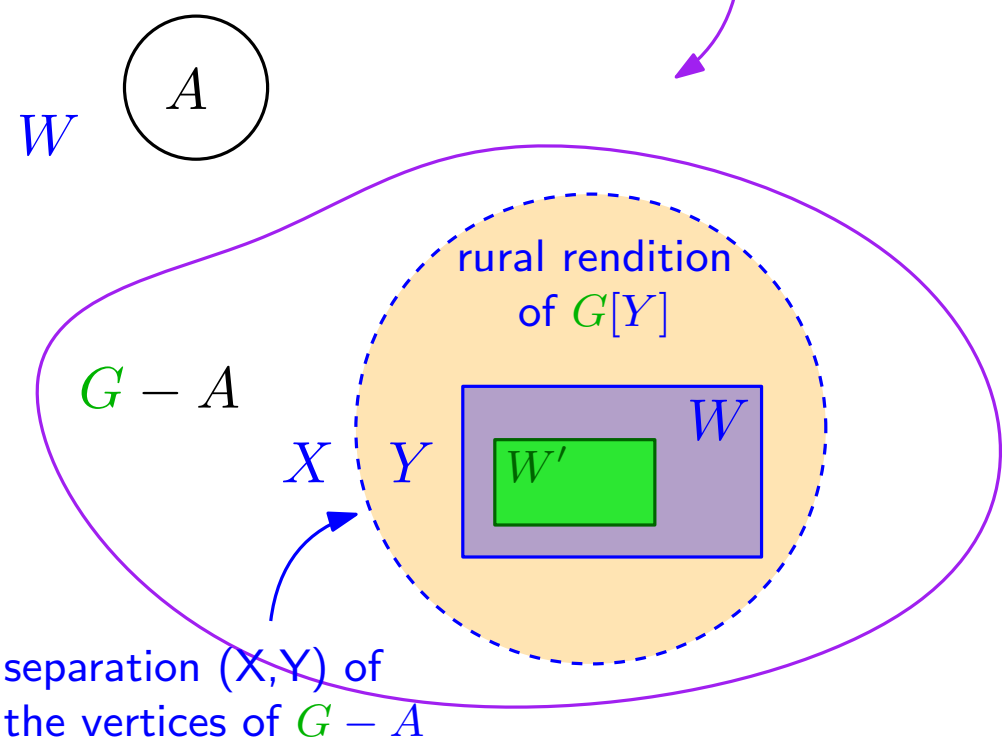
Solution:

find a **homogeneous wall** W' inside W

Folio of a cell $c \approx$ all minors present in $c + A$ of small size that are rooted at boundary$(c) + A$



folio $\approx$ all patterns in here



Far more difficult case: kDPP on general graphs

[Graph Minors XIII]

Flat Wall theorem

FPT algo in k and t

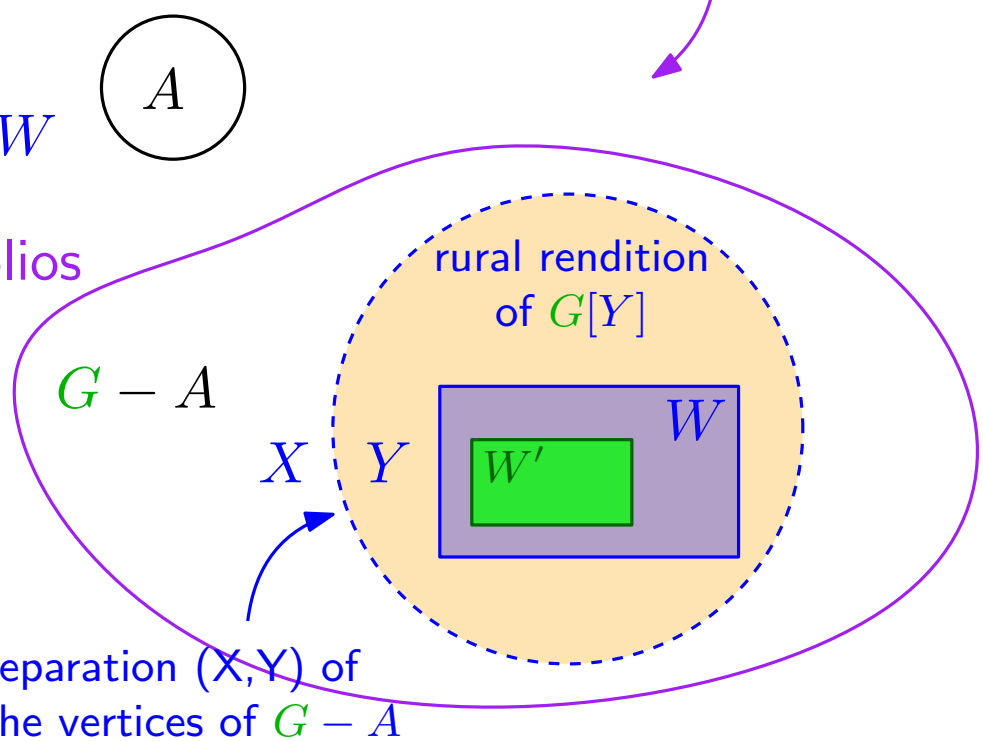
A graph G either has:

- treewidth at most $f(k,t)$, or
- a clique of size t as a minor, or
- a vertex set $A \leq g(t)$ and a wall W of height k s.t. W is a **flat wall** of $G - A$.

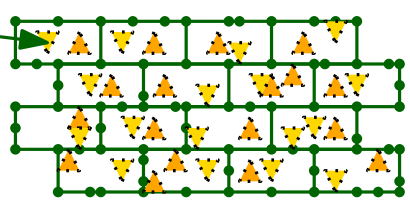
Solution:

find a **homogeneous wall** W' inside W

each brick of W' contains the same folios



a brick



Far more difficult case: kDPP on general graphs

[Graph Minors XIII]

Flat Wall theorem

FPT algo in k and t

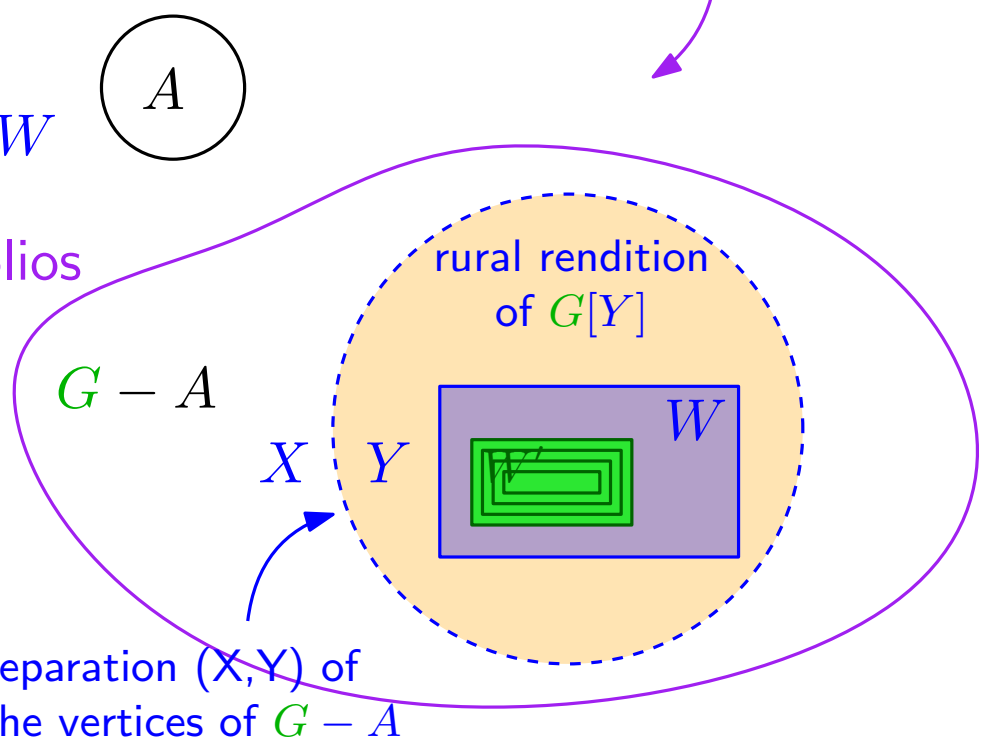
A graph G either has:

- treewidth at most $f(k,t)$, or
- a clique of size t as a minor, or
- a vertex set $A \leq g(t)$ and a wall W of height k s.t. W is a **flat wall** of $G - A$.

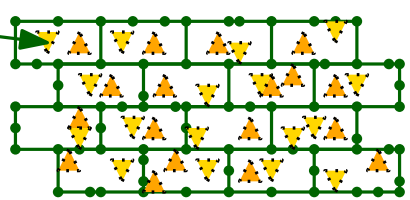
Solution:

find a **homogeneous wall** W' inside W

each brick of W' contains the same folios



a brick



Far more difficult case: kDPP on general graphs

[Graph Minors XIII]

Flat Wall theorem

FPT algo in k and t

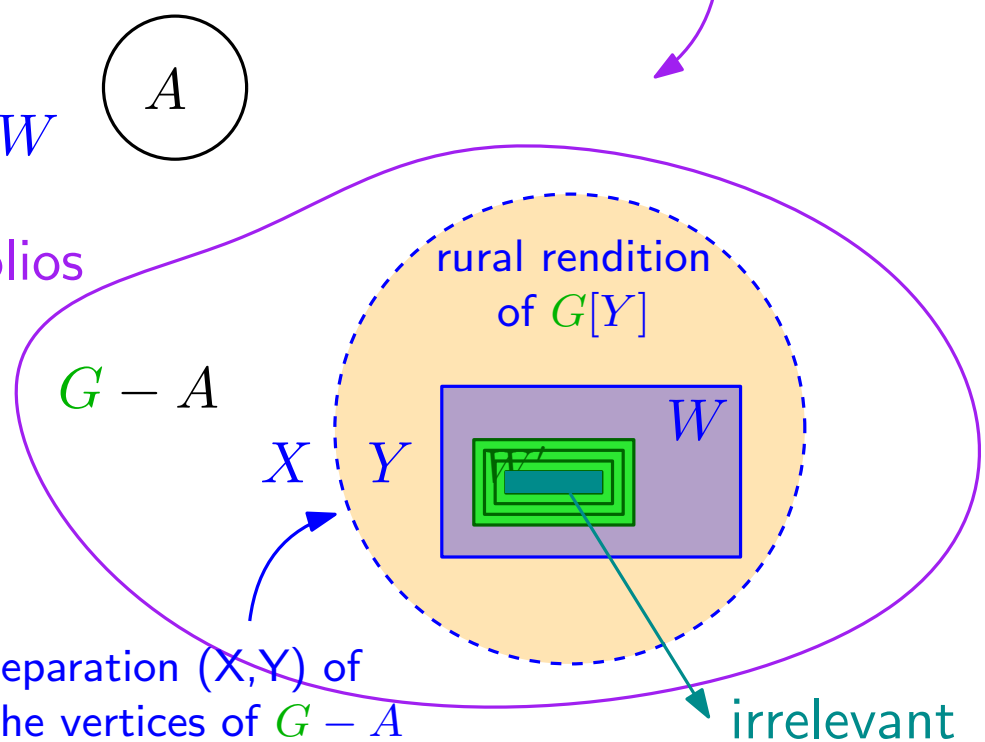
A graph G either has:

- treewidth at most $f(k,t)$, or
- a clique of size t as a minor, or
- a vertex set $A \leq g(t)$ and a wall W of height k s.t. W is a **flat wall** of $G - A$.

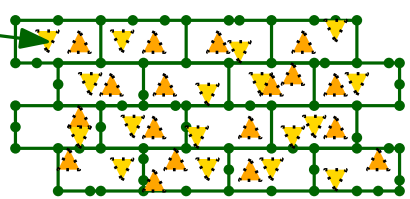
Solution:

find a **homogeneous wall** W' inside W

each brick of W' contains the same folios



a brick



Far more difficult case: kDPP on general graphs

[Graph Minors XIII]

Flat Wall theorem

FPT algo in k and t

A graph G either has:

- treewidth at most $f(k,t)$, or
- a clique of size t as a minor, or
- a vertex set $A \leq g(t)$ and a wall W of height k s.t. W is a **flat wall** of $G - A$.

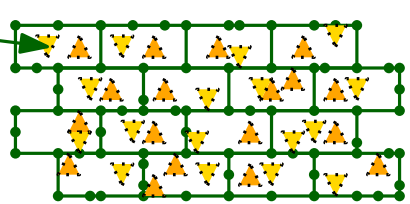
Solution:

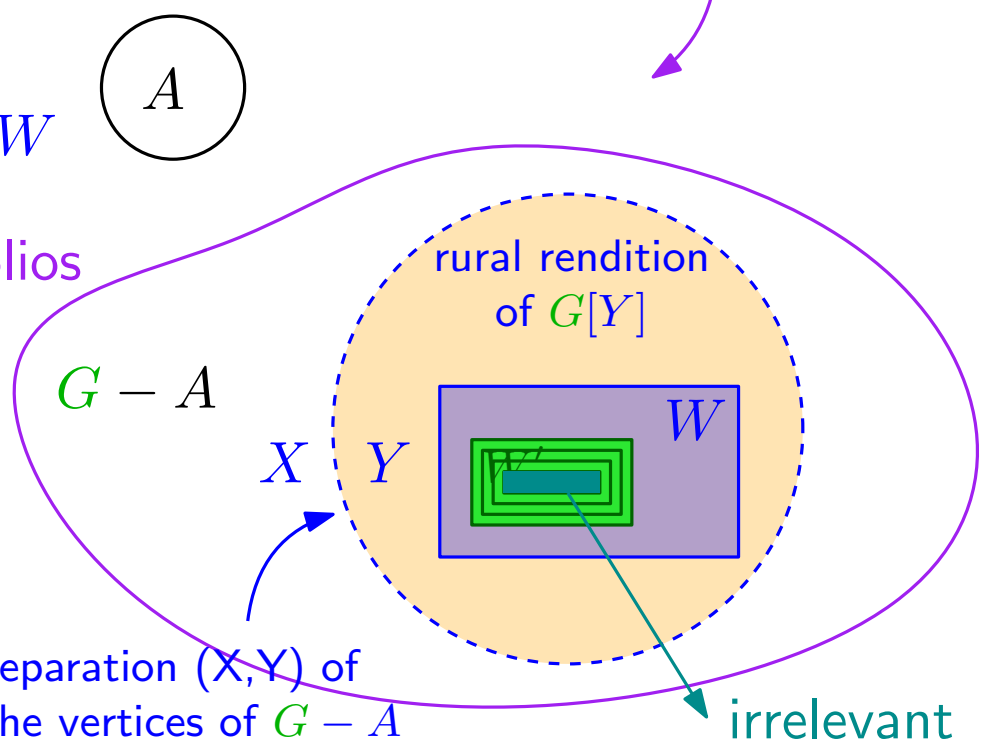
find a **homogeneous wall** W' inside W

each brick of W' contains the same folios

$\Longrightarrow$ can delete the central part of W' and recurse.

a brick





[Graph Minors XIII]

kDPP is solvable in time $f(k) \cdot n^3$.

[Graph Minors XIII]

kDPP is solvable in time $f(k) \cdot n^3$.

More generally, they prove (with the same technique):

[Graph Minors XIII]

One can check whether a graph H is a minor of G in time $f(|V(H)|) \cdot n^3$.

[Graph Minors XIII]

kDPP is solvable in time $f(k) \cdot n^3$.

More generally, they prove (with the same technique):

[Graph Minors XIII] 1995

One can check whether a graph H is a minor of G in time $f(|V(H)|) \cdot n^3$.

Improvements on the running time:

[Kawarabayashi, Kobayashi, Reed, 2012] $f(|V(H)|) \cdot n^2$

[Korhonen, Pilipczuk, Stamoulis, 2024] $f(|V(H)|) \cdot n^{1+o(1)}$

[Graph Minors XIII]
kDPP is solvable in time $f(k) \cdot n^3$.

More generally, they prove (with the same technique):

[Graph Minors XIII] 1995
One can check whether a graph H is a minor of G in time $f(|V(H)|) \cdot n^3$.

Improvements on the running time:

[Kawarabayashi, Kobayashi, Reed, 2012] $f(|V(H)|) \cdot n^2$
[Korhonen, Pilipczuk, Stamoulis, 2024] $f(|V(H)|) \cdot n^{1+o(1)}$

Big open problem: $f(|V(H)|) \cdot n$?

Goal of the Graph Minor series:

Wagner's conjecture

[Graph Minors XX]

Wagner's conjecture [Wagner, 1937]

Graphs are **well quasi ordered** under the minor relation.

[Graph Minors XX] 2004

Graphs are **well quasi ordered** under the minor relation.

Wagner's conjecture [Wagner, 1937]

Graphs are **well quasi ordered** under the minor relation.

[Graph Minors XX] 2004

Graphs are **well quasi ordered** under the minor relation.

What does it mean and why is it so important?

A **quasi order** $\preceq$ is a binary relation that is both reflexive ($x \preceq x$) and transitive ($x \preceq y$ and $y \preceq z \implies x \preceq z$)

$\implies$ Minor containment defines a quasi ordering on graphs:
if $G_1 \preceq_{\mathsf{m}} G_2$ and $G_2 \preceq_{\mathsf{m}} G_3$, then $G_1 \preceq_{\mathsf{m}} G_3$

An **antichain** is a set of pairwise non-comparable elements ($x \not\preceq y$ and $y \not\preceq x$).

A quasi order $\preceq$ defined on a set X is a **well quasi order** (**WQO**) if there is no infinite strictly decreasing sequence and no infinite antichain.

Are those WQO?

- $(\mathbb{N}, \leq)$
- $(\mathbb{R}, \leq)$
- $(\mathbb{N}^2, \preceq)$, where $(x, y) \preceq (x', y')$ iff ($x \leq x'$ and $y \leq y'$)
- graphs under the induced subgraph relation (delete vertices)
- graphs under the subgraph relation (delete vertices and edges)

A **quasi order** $\preceq$ is a binary relation that is both reflexive ($x \preceq x$) and transitive ($x \preceq y$ and $y \preceq z \implies x \preceq z$)

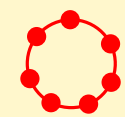
$\implies$ Minor containment defines a quasi ordering on graphs:
if $G_1 \preceq_{\mathsf{m}} G_2$ and $G_2 \preceq_{\mathsf{m}} G_3$, then $G_1 \preceq_{\mathsf{m}} G_3$

An **antichain** is a set of pairwise non-comparable elements ($x \not\preceq y$ and $y \not\preceq x$).

A quasi order $\preceq$ defined on a set X is a **well quasi order** (**WQO**) if there is no infinite strictly decreasing sequence and no infinite antichain.

Are those WQO?

- $(\mathbb{N}, \leq)$ — yes
- $(\mathbb{R}, \leq)$ — no
- $(\mathbb{N}^2, \preceq)$, where $(x, y) \preceq (x', y')$ iff ($x \leq x'$ and $y \leq y'$) — yes
- graphs under the induced subgraph relation (delete vertices) — no
- graphs under the subgraph relation (delete vertices and edges) — no



A quasi order $\preceq$ defined on a set X is a **well quasi order** (**WQO**) if there is no infinite strictly decreasing sequence and no infinite antichain.

[Graph Minors XX]

Graphs are **well quasi ordered** under the minor relation.

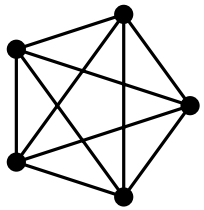
A quasi order $\preceq$ defined on a set X is a **well quasi order** (**WQO**) if there is no infinite strictly decreasing sequence and no infinite antichain.

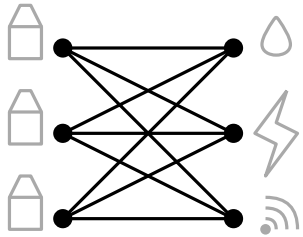
[Graph Minors XX]

Graphs are **well quasi ordered** under the minor relation.

An **obstruction** of a minor-closed graph class $\mathcal{H}$ is a graph F that do not belong to $\mathcal{H}$, but whose minors all belong to $\mathcal{H}$.

$\Longrightarrow$ Obs(planar) $= \{K_5, K_{3,3}\}$





The obstructions of $\mathcal{H}$ are pairwise non-comparable (under the minor relation).

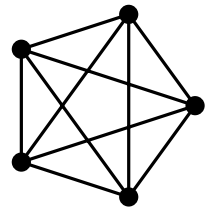
$\Longrightarrow$ Obs($\mathcal{H}$) is an antichain.

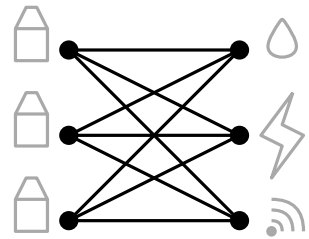
A quasi order $\preceq$ defined on a set X is a **well quasi order** (**WQO**) if there is no infinite strictly decreasing sequence and no infinite antichain.

[Graph Minors XX]
Graphs are **well quasi ordered** under the minor relation.

An **obstruction** of a minor-closed graph class $\mathcal{H}$ is a graph F that do not belong to $\mathcal{H}$, but whose minors all belong to $\mathcal{H}$.

$\Longrightarrow$ Obs(planar) $= \{K_5, K_{3,3}\}$





The obstructions of $\mathcal{H}$ are pairwise non-comparable (under the minor relation).

$\Longrightarrow$ Obs($\mathcal{H}$) is an antichain.

Graph Minor theorem [Graph Minors XX]
A minor-closed graph class $\mathcal{H}$ has a finite number of obstructions.

Graph Minor theorem [Graph Minors XX]

A minor-closed graph class $\mathcal{H}$ has a finite number of obstructions.

$+$

Minor containment [Graph Minors XIII]

One can check whether a graph H is a minor of G in time $f(|V(H)|) \cdot n^3$.

$=$

Membership to $\mathcal{H}$

One can check whether a graph belongs to a minor-closed graph class $\mathcal{H}$ in time $\mathcal{O}_{\mathcal{H}}(n^3)$.

Proof

- List all obstructions of $\mathcal{H}$ (finite number ℓ): $F_1, \ldots, F_\ell$.
- for $1 \leq i \leq \ell$, check whether F_i is a minor of G.
- If there is i s.t. F_i is a minor of G, then $G \notin \mathcal{H}$.
- Otherwise, $G \in \mathcal{H}$.

Graph Minor theorem [Graph Minors XX]

A minor-closed graph class $\mathcal{H}$ has a finite number of obstructions.

$+$

Minor containment [Graph Minors XIII]

One can check whether a graph H is a minor of G in time $f(|V(H)|) \cdot n^3$.

$n^{1+o(1)}$

$=$

[Korhonen, Pilipczuk, Stamoulis, 2024]

Membership to $\mathcal{H}$

One can check whether a graph belongs to a minor-closed graph class $\mathcal{H}$ in time $\mathcal{O}_{\mathcal{H}}(n^3)$.

$n^{1+o(1)}$

Proof

- List all obstructions of $\mathcal{H}$ (finite number ℓ): $F_1, \ldots, F_\ell$.
- for $1 \leq i \leq \ell$, check whether F_i is a minor of G.
- If there is i s.t. F_i is a minor of G, then $G \notin \mathcal{H}$.
- Otherwise, $G \in \mathcal{H}$.

Graph Minor theorem [Graph Minors XX]

A minor-closed graph class $\mathcal{H}$ has a finite number of obstructions.

$+$

Minor containment [Graph Minors XIII]

One can check whether a graph H is a minor of G in time $f(|V(H)|) \cdot n^3$.

$n^{1+o(1)}$

$=$

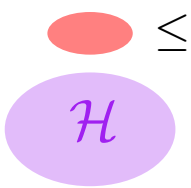
[Korhonen, Pilipczuk, Stamoulis, 2024]

Membership to $\mathcal{H}$

One can check whether a graph belongs to a minor-closed graph class $\mathcal{H}$ in time $\mathcal{O}_{\mathcal{H}}(n^3)$.

$n^{1+o(1)}$

Application:



VERTEX DELETION TO $\mathcal{H}$

Input: A graph G, $k \in \mathbb{N}$.

Question: Is there a set S of size $\leq k$ s.t. $G - S \in \mathcal{H}$?

- $\mathcal{H}$= edgeless $\rightarrow$ VERTEX COVER
- $\mathcal{H}$= forest $\rightarrow$ FEEDBACK VERTEX SET
- $\mathcal{H}$= planar $\rightarrow$ PLANARIZATION

Graph Minor theorem [Graph Minors XX]

A minor-closed graph class $\mathcal{H}$ has a finite number of obstructions.

$+$

Minor containment [Graph Minors XIII]

One can check whether a graph H is a minor of G in time $f(|V(H)|) \cdot n^3$.

$n^{1+o(1)}$

$=$

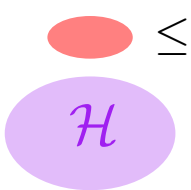
[Korhonen, Pilipczuk, Stamoulis, 2024]

Membership to $\mathcal{H}$

One can check whether a graph belongs to a minor-closed graph class $\mathcal{H}$ in time $\mathcal{O}_{\mathcal{H}}(n^3)$.

$n^{1+o(1)}$

Application:



VERTEX DELETION TO $\mathcal{H}$

Input: A graph G, $k \in \mathbb{N}$.

Question: Is there a set S of size $\leq k$ s.t. $G - S \in \mathcal{H}$?

$\mathcal{G}_{\mathcal{H},k}$ = graphs that belong to the minor-closed graph class $\mathcal{H}$ after deleting $\leq k$ vertices

minor-closed graph class

Graph Minor theorem [Graph Minors XX]

A minor-closed graph class $\mathcal{H}$ has a finite number of obstructions.

$+$

Minor containment [Graph Minors XIII]

One can check whether a graph H is a minor of G in time $f(|V(H)|) \cdot n^3$.

$n^{1+o(1)}$

$=$

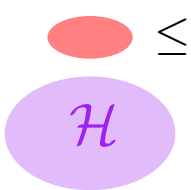
[Korhonen, Pilipczuk, Stamoulis, 2024]

Membership to $\mathcal{H}$

One can check whether a graph belongs to a minor-closed graph class $\mathcal{H}$ in time $\mathcal{O}_{\mathcal{H}}(n^3)$.

$n^{1+o(1)}$

Application:



VERTEX DELETION TO $\mathcal{H}$

Input: A graph G, $k \in \mathbb{N}$.

Question: Is there a set S of size $\leq k$ s.t. $G - S \in \mathcal{H}$?

$\mathcal{G}_{\mathcal{H},k}$ = graphs that belong to the minor-closed graph class $\mathcal{H}$ after deleting $\leq k$ vertices

minor-closed graph class $=$ yes-instances of VERTEX DELETION TO $\mathcal{H}$ (for a fixed k)

Graph Minor theorem [Graph Minors XX]

A minor-closed graph class $\mathcal{H}$ has a finite number of obstructions.

$+$

Minor containment [Graph Minors XIII]

One can check whether a graph H is a minor of G in time $f(|V(H)|) \cdot n^3$.

$n^{1+o(1)}$

$=$

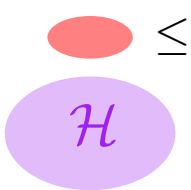
[Korhonen, Pilipczuk, Stamoulis, 2024]

Membership to $\mathcal{H}$

One can check whether a graph belongs to a minor-closed graph class $\mathcal{H}$ in time $\mathcal{O}_{\mathcal{H}}(n^3)$.

$n^{1+o(1)}$

Application:



VERTEX DELETION TO $\mathcal{H}$

Input: A graph G, $k \in \mathbb{N}$.

Question: Is there a set S of size $\leq k$ s.t. $G - S \in \mathcal{H}$?

$\mathcal{G}_{\mathcal{H},k}$ = graphs that belong to the minor-closed graph class $\mathcal{H}$ after deleting $\leq k$ vertices

minor-closed graph class = yes-instances of VERTEX DELETION TO $\mathcal{H}$ (for a fixed k)

$\Longrightarrow$ VERTEX DELETION TO $\mathcal{H}$ is solvable in time $f(k) \cdot n^{1+o(1)}$.

Conclusion

Graph Minor theory is **great** because

- many problem related to minor-closed graph classes can be solved efficiently
- many powerful and very general techniques have been developped to solve problems in Graph Minor theory

Conclusion

Graph Minor theory is **great** because

- many problem related to minor-closed graph classes can be solved efficiently
- many powerful and very general techniques have been developped to solve problems in Graph Minor theory

... **but**

- requires lots of definitions that are not so easy to understand nor to use
- From Wikipedia, the free encyclopedia

  A **galactic algorithm** is an algorithm with record-breaking theoretical (asymptotic) performance, but which is not used due to practical constraints. Typical reasons are that the performance gains only appear for problems that are so large they never occur, or the algorithm's complexity outweighs a relatively small gain in real-world performance. Galactic algorithms were so named by Richard Lipton and Ken Regan,[1] because they will never be used on any data sets on Earth.

  Sub-graphs [edit]

  The problem of deciding whether a graph G contains H as a minor is NP-complete in general, but where H is fixed, it can be solved in polynomial time. The running time for testing whether H is a minor of G in this case is $O(n^2)$,[9] where n is the number of vertices in G and the big O notation hides a constant that depends superexponentially on H. The constant is greater than $2 \uparrow\uparrow (2 \uparrow\uparrow (2 \uparrow\uparrow (h/2)))$ in Knuth's up-arrow notation, where h is the number of vertices in H.[10] Even the case of $h = 4$ cannot be reasonably computed as the constant is greater than 2 pentated by 4, or 2 tetrated by 65536, that is, $2 \uparrow\uparrow\uparrow 4 = {}^{65536}2 = \underbrace{2^{2^{\cdot^{\cdot^{2}}}}}_{65536}$.

Conclusion

Graph Minor theory is **great** because

- many problem related to minor-closed graph classes can be solved efficiently
- many powerful and very general techniques have been developped to solve problems in Graph Minor theory

The End.

Conclusion

Graph Minor theory is **great** because

- many problem related to minor-closed graph classes can be solved efficiently
- many powerful and very general techniques have been developped to solve problems in Graph Minor theory

Questions?

The End.