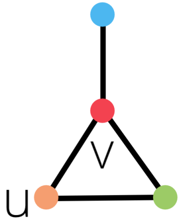
Vertex-minors and flooding immersions

Rose McCarty
Joint work with Jim Geelen and Paul Wollan (ongoing!)

IBS Virtual Discrete Math Colloquium
January 2021

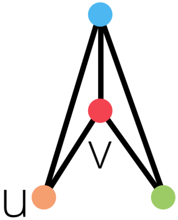


G

Locally complementing at v replaces the induced subgraph on the neighbourhood of v by its complement.

Two graphs are **locally equivalent** if one can be obtained from the other by local complementations.

A graph H is a **vertex-minor** of G if H can be obtained from a graph that is locally equivalent to G by deleting vertices.

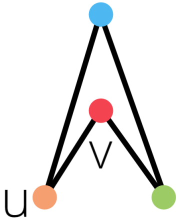


$G * v$

Locally complementing at v replaces the induced subgraph on the neighbourhood of v by its complement.

Two graphs are **locally equivalent** if one can be obtained from the other by local complementations.

A graph H is a **vertex-minor** of G if H can be obtained from a graph that is locally equivalent to G by deleting vertices.

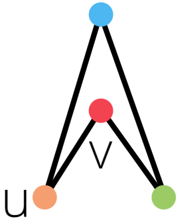


$$G * v * u$$

Locally complementing at v replaces the induced subgraph on the neighbourhood of v by its complement.

Two graphs are **locally equivalent** if one can be obtained from the other by local complementations.

A graph H is a **vertex-minor** of G if H can be obtained from a graph that is locally equivalent to G by deleting vertices.

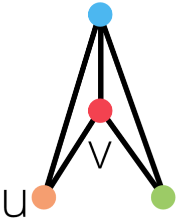


$$G * v * u$$

Locally complementing at v replaces the induced subgraph on the neighbourhood of v by its complement.

Two graphs are **locally equivalent** if one can be obtained from the other by local complementations.

A graph H is a **vertex-minor** of G if H can be obtained from a graph that is locally equivalent to G by deleting vertices.

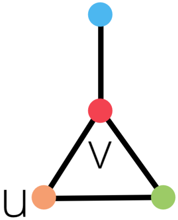


$$G * v * u * u$$

Locally complementing at v replaces the induced subgraph on the neighbourhood of v by its complement.

Two graphs are **locally equivalent** if one can be obtained from the other by local complementations.

A graph H is a **vertex-minor** of G if H can be obtained from a graph that is locally equivalent to G by deleting vertices.

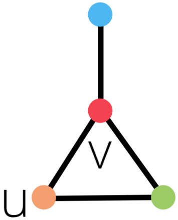


$$G * v * u * u * v =$$

Locally complementing at v replaces the induced subgraph on the neighbourhood of v by its complement.

Two graphs are **locally equivalent** if one can be obtained from the other by local complementations.

A graph H is a **vertex-minor** of G if H can be obtained from a graph that is locally equivalent to G by deleting vertices.

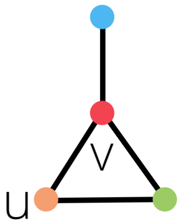


G

Locally complementing at v replaces the induced subgraph on the neighbourhood of v by its complement.

Two graphs are **locally equivalent** if one can be obtained from the other by local complementations.

A graph H is a **vertex-minor** of G if H can be obtained from a graph that is locally equivalent to G by deleting vertices.

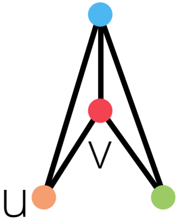


G

Locally complementing at v replaces the induced subgraph on the neighbourhood of v by its complement.

Two graphs are **locally equivalent** if one can be obtained from the other by local complementations.

A graph H is a **vertex-minor** of G if H can be obtained from a graph that is locally equivalent to G by deleting vertices.

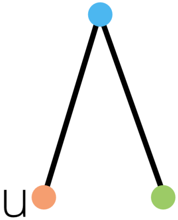


$$G * v$$

Locally complementing at v replaces the induced subgraph on the neighbourhood of v by its complement.

Two graphs are **locally equivalent** if one can be obtained from the other by local complementations.

A graph H is a **vertex-minor** of G if H can be obtained from a graph that is locally equivalent to G by deleting vertices.



$$G * v - v$$

Locally complementing at v replaces the induced subgraph on the neighbourhood of v by its complement.

Two graphs are **locally equivalent** if one can be obtained from the other by local complementations.

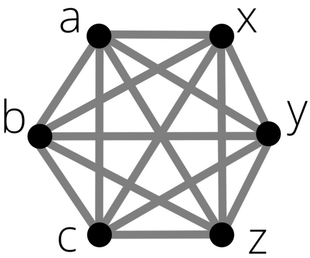
A graph H is a **vertex-minor** of G if H can be obtained from a graph that is locally equivalent to G by deleting vertices.

Why **local equivalence**?

- Cut-rank: "connectivity for dense graphs" [Bouchet; Oum 05]
- Graph states: "resources in quantum computing" [Raussendorf-Briegel 01, Van den Nest-Dehaene-De Moor 04]
- Delta-matroids: "matroid theory for symmetric matrices" [Bouchet; Moffatt 19]

Why **local equivalence**?

- Cut-rank: “connectivity for dense graphs” [Bouchet; Oum 05]
- Graph states: “resources in quantum computing” [Raussendorf-Briegel 01, Van den Nest-Dehaene-De Moor 04]
- Delta-matroids: “matroid theory for symmetric matrices” [Bouchet; Moffatt 19]



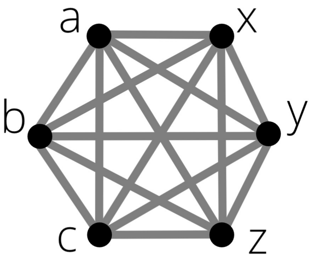
Why **local equivalence**?

- Cut-rank: “connectivity for dense graphs” [Bouchet; Oum 05]
- Graph states: “resources in quantum computing” [Raussendorf-Briegel 01, Van den Nest-Dehaene-De Moor 04]
- Delta-matroids: “matroid theory for symmetric matrices” [Bouchet; Moffatt 19]

$$\begin{array}{c|ccc} & x & y & z \\ \hline a & 1 & 1 & 1 \\ b & 1 & 1 & 1 \\ c & 1 & 1 & 1 \end{array}$$

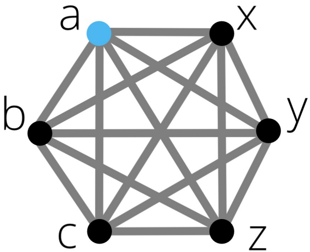
Why **local equivalence**?

- Cut-rank: "connectivity for dense graphs" [Bouchet; Oum 05]
- Graph states: "resources in quantum computing" [Raussendorf-Briegel 01, Van den Nest-Dehaene-De Moor 04]
- Delta-matroids: "matroid theory for symmetric matrices" [Bouchet; Moffatt 19]



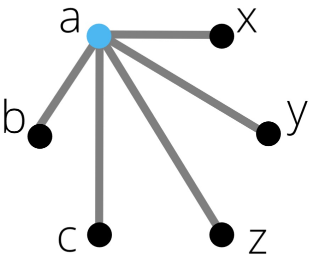
Why **local equivalence**?

- Cut-rank: “connectivity for dense graphs” [Bouchet; Oum 05]
- Graph states: “resources in quantum computing” [Raussendorf-Briegel 01, Van den Nest-Dehaene-De Moor 04]
- Delta-matroids: “matroid theory for symmetric matrices” [Bouchet; Moffatt 19]



Why **local equivalence**?

- Cut-rank: “connectivity for dense graphs” [Bouchet; Oum 05]
- Graph states: “resources in quantum computing” [Raussendorf-Briegel 01, Van den Nest-Dehaene-De Moor 04]
- Delta-matroids: “matroid theory for symmetric matrices” [Bouchet; Moffatt 19]



Why **local equivalence**?

- Cut-rank: "connectivity for dense graphs" [Bouchet; Oum 05]
- Graph states: "resources in quantum computing" [Raussendorf-Briegel 01, Van den Nest-Dehaene-De Moor 04]
- Delta-matroids: "matroid theory for symmetric matrices" [Bouchet; Moffatt 19]

$$\begin{array}{cc} & \begin{array}{ccc} x & y & z \end{array} \\ \begin{array}{c} a \\ b \\ c \end{array} & \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \end{array}$$

Why **local equivalence**?

- Cut-rank: "connectivity for dense graphs" [Bouchet; Oum 05]
- Graph states: "resources in quantum computing" [Raussendorf-Briegel 01, Van den Nest-Dehaene-De Moor 04]
- Delta-matroids: "matroid theory for symmetric matrices" [Bouchet; Moffatt 19]

$$\begin{array}{c|ccc} & x & y & z \\ \hline a & 1 & 1 & 1 \\ b & 1 & 1 & 1 \\ c & 1 & 1 & 1 \end{array}$$

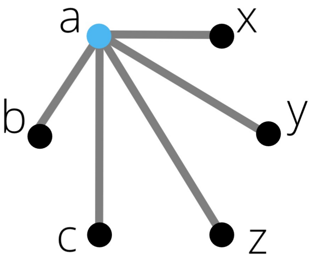
Why **local equivalence**?

- Cut-rank: "connectivity for dense graphs" [Bouchet; Oum 05]
- Graph states: "resources in quantum computing" [Raussendorf-Briegel 01, Van den Nest-Dehaene-De Moor 04]
- Delta-matroids: "matroid theory for symmetric matrices" [Bouchet; Moffatt 19]

$$\begin{array}{c|ccc} & x & y & z \\ \hline a & 1 & 1 & 1 \\ b & 0 & 0 & 0 \\ c & 0 & 0 & 0 \end{array}$$

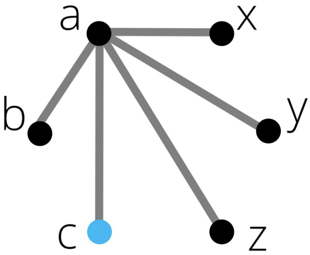
Why **local equivalence**?

- Cut-rank: “connectivity for dense graphs” [Bouchet; Oum 05]
- Graph states: “resources in quantum computing” [Raussendorf-Briegel 01, Van den Nest-Dehaene-De Moor 04]
- Delta-matroids: “matroid theory for symmetric matrices” [Bouchet; Moffatt 19]



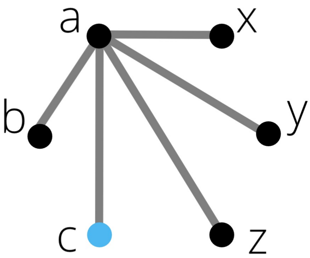
Why **local equivalence**?

- Cut-rank: "connectivity for dense graphs" [Bouchet; Oum 05]
- Graph states: "resources in quantum computing" [Raussendorf-Briegel 01, Van den Nest-Dehaene-De Moor 04]
- Delta-matroids: "matroid theory for symmetric matrices" [Bouchet; Moffatt 19]



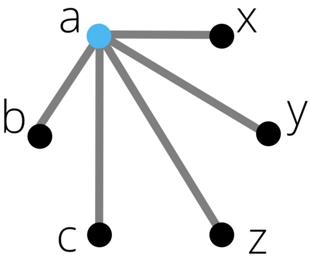
Why **local equivalence**?

- Cut-rank: "connectivity for dense graphs" [Bouchet; Oum 05]
- Graph states: "resources in quantum computing" [Raussendorf-Briegel 01, Van den Nest-Dehaene-De Moor 04]
- Delta-matroids: "matroid theory for symmetric matrices" [Bouchet; Moffatt 19]



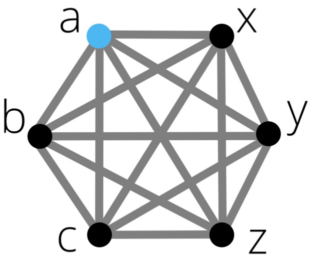
Why **local equivalence**?

- Cut-rank: "connectivity for dense graphs" [Bouchet; Oum 05]
- Graph states: "resources in quantum computing" [Raussendorf-Briegel 01, Van den Nest-Dehaene-De Moor 04]
- Delta-matroids: "matroid theory for symmetric matrices" [Bouchet; Moffatt 19]



Why **local equivalence**?

- Cut-rank: "connectivity for dense graphs" [Bouchet; Oum 05]
- Graph states: "resources in quantum computing" [Raussendorf-Briegel 01, Van den Nest-Dehaene-De Moor 04]
- Delta-matroids: "matroid theory for symmetric matrices" [Bouchet; Moffatt 19]



Why **local equivalence**?

- Cut-rank: "connectivity for dense graphs" [Bouchet; Oum 05]
- Graph states: "resources in quantum computing" [Raussendorf-Briegel 01, Van den Nest-Dehaene-De Moor 04]
- Delta-matroids: "matroid theory for symmetric matrices" [Bouchet; Moffatt 19]

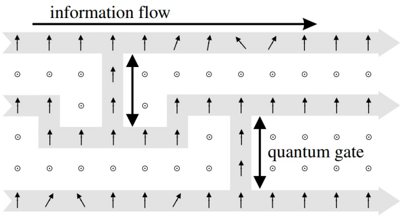


FIG. 1. Quantum computation by measuring two-state parti-

Why **local equivalence**?

- Cut-rank: "connectivity for dense graphs" [Bouchet; Oum 05]
- Graph states: "resources in quantum computing" [Raussendorf-Briegel 01, Van den Nest-Dehaene-De Moor 04]
- Delta-matroids: "matroid theory for symmetric matrices" [Bouchet; Moffatt 19]

$$\begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

Why **local equivalence**?

- Cut-rank: "connectivity for dense graphs" [Bouchet; Oum 05]
- Graph states: "resources in quantum computing" [Raussendorf-Briegel 01, Van den Nest-Dehaene-De Moor 04]
- Delta-matroids: "matroid theory for symmetric matrices" [Bouchet; Moffatt 19]

Can we describe the structure of graphs with no **vertex-minor** isomorphic to H?

Why **local equivalence**?

- Cut-rank: "connectivity for dense graphs" [Bouchet; Oum 05]
- Graph states: "resources in quantum computing" [Raussendorf-Briegel 01, Van den Nest-Dehaene-De Moor 04]
- Delta-matroids: "matroid theory for symmetric matrices" [Bouchet; Moffatt 19]

Can we describe the structure of graphs with no **vertex-minor** isomorphic to H?

- Have pure pairs of size $\epsilon_H \cdot n$ [Chudnovsky-Oum 18]
- Have chromatic number $\leq f_H(\text{clique number})$ [Davies 21]

Why **local equivalence**?

- Cut-rank: “connectivity for dense graphs” [Bouchet; Oum 05]
- Graph states: “resources in quantum computing” [Raussendorf-Briegel 01, Van den Nest-Dehaene-De Moor 04]
- Delta-matroids: “matroid theory for symmetric matrices” [Bouchet; Moffatt 19]

Can we describe the structure of graphs with no **vertex-minor** isomorphic to H?

- Have pure pairs of size $\epsilon_H \cdot n$ [Chudnovsky-Oum 18]
- Have chromatic number $\leq f_H$(clique number) [Davies 21]

Why **local equivalence**?

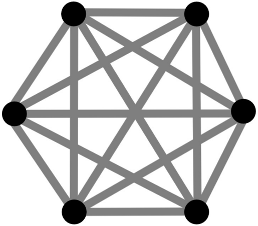
- Cut-rank: “connectivity for dense graphs” [Bouchet; Oum 05]
- Graph states: “resources in quantum computing” [Raussendorf-Briegel 01, Van den Nest-Dehaene-De Moor 04]
- Delta-matroids: “matroid theory for symmetric matrices” [Bouchet; Moffatt 19]

Can we describe the structure of graphs with no **vertex-minor** isomorphic to H?

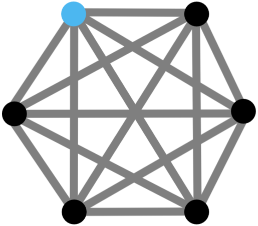
- Have pure pairs of size $\epsilon_H \cdot n$ [Chudnovsky-Oum 18]
- Have chromatic number $\leq f_H$(clique number) [Davies 21]

Want a structure that guarantees some H' is not a vertex-minor.

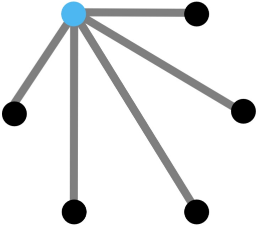
Difficulty: Our graph class can have arbitrarily large cliques.



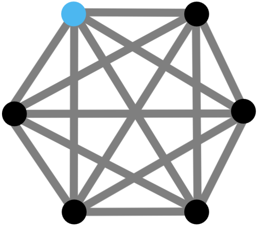
Difficulty: Our graph class can have arbitrarily large cliques.



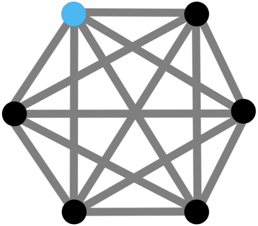
Difficulty: Our graph class can have arbitrarily large cliques.



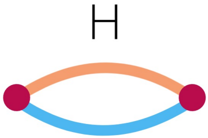
Difficulty: Our graph class can have arbitrarily large cliques.

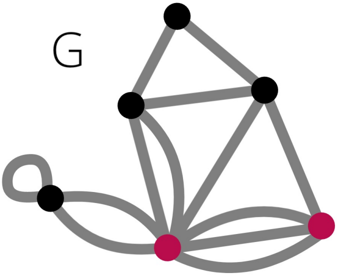


Difficulty: Our graph class can have arbitrarily large cliques.



Approach: Kotzig and Bouchet found a connection with **flooding immersions**.

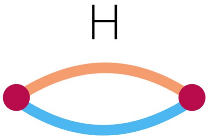


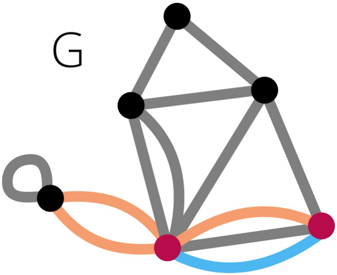


An **immersion** of H into G consists of

- an injection $\psi : V(H) \to V(G)$ and
- for each $e = uv \in E(H)$, a $(\psi(u), \psi(v))$-trail in G, s.t. the trails are edge-disjoint.

It is **flooding** if every edge of G is in one of the trails; we say H **floods** G. If H and G are Eulerian, every immersion can be made flooding.

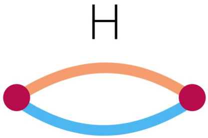


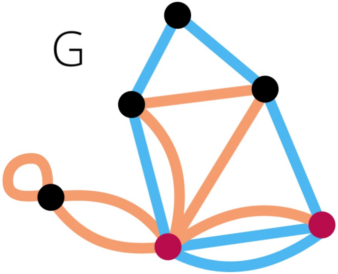


An **immersion** of H into G consists of

- an injection $\psi : V(H) \to V(G)$ and
- for each $e = uv \in E(H)$, a $(\psi(u), \psi(v))$-trail in G, s.t. the trails are edge-disjoint.

It is **flooding** if every edge of G is in one of the trails; we say H **floods** G. If H and G are Eulerian, every immersion can be made flooding.

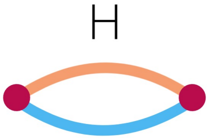


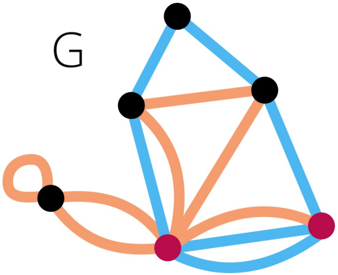


An **immersion** of H into G consists of

- an injection $\psi : V(H) \to V(G)$ and
- for each $e = uv \in E(H)$, a $(\psi(u), \psi(v))$-trail in G, s.t. the trails are edge-disjoint.

It is **flooding** if every edge of G is in one of the trails; we say H **floods** G. If H and G are Eulerian, every immersion can be made flooding.

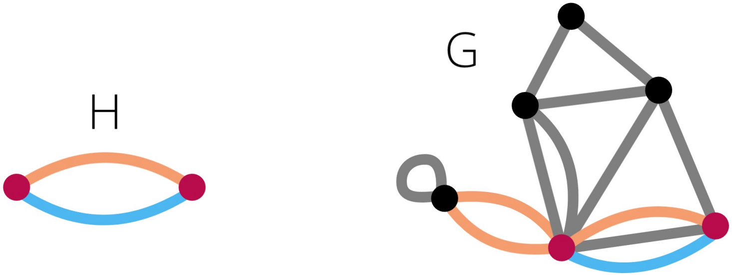




An **immersion** of H into G consists of

- an injection $\psi : V(H) \to V(G)$ and
- for each $e = uv \in E(H)$, a $(\psi(u), \psi(v))$-trail in G, s.t. the trails are edge-disjoint.

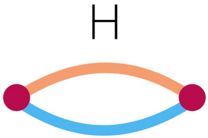
It is **flooding** if every edge of G is in one of the trails; we say H **floods** G. If H and G are Eulerian, every immersion can be made flooding.

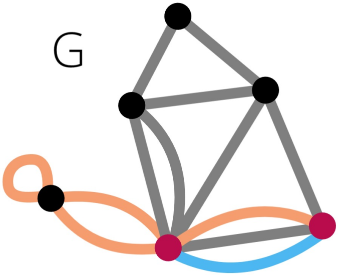


An **immersion** of H into G consists of

- an injection $\psi : V(H) \to V(G)$ and
- for each $e = uv \in E(H)$, a $(\psi(u), \psi(v))$-trail in G, s.t. the trails are edge-disjoint.

It is **flooding** if every edge of G is in one of the trails; we say H **floods** G. If H and G are Eulerian, every immersion can be made flooding.

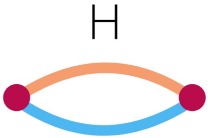


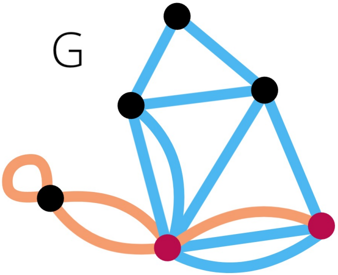


An **immersion** of H into G consists of

- an injection $\psi : V(H) \to V(G)$ and
- for each $e = uv \in E(H)$, a $(\psi(u), \psi(v))$-trail in G, s.t. the trails are edge-disjoint.

It is **flooding** if every edge of G is in one of the trails; we say H **floods** G. If H and G are Eulerian, every immersion can be made flooding.

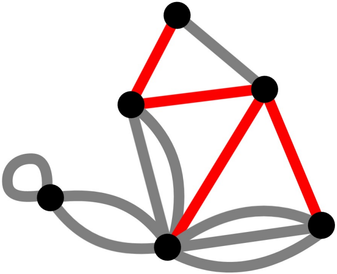




An **immersion** of H into G consists of

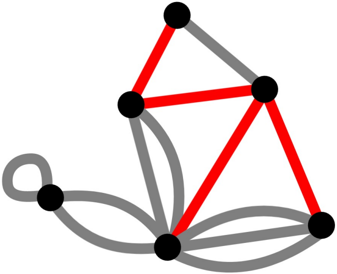
- an injection $\psi : V(H) \rightarrow V(G)$ and
- for each $e = uv \in E(H)$, a $(\psi(u), \psi(v))$-trail in G, s.t. the trails are edge-disjoint.

It is **flooding** if every edge of G is in one of the trails; we say H **floods** G. If H and G are Eulerian, every immersion can be made flooding.



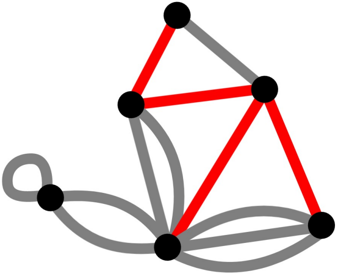
A **signed graph** is a graph with a specified set Σ of edges, called the **signature**. Equivalently, each edge e has a **weight** $w(e) \in \mathbb{Z}_2$.

We only care about the **weight** of each cycle, where $w(C) := \sum_{e \in E(C)} w(e)$ over $\mathbb{Z}_2$. So **re-signing** (adding 1 to each edge in a cut) gives an equivalent signed graph.



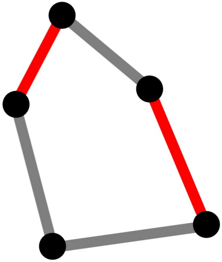
A **signed graph** is a graph with a specified set Σ of edges, called the **signature**. Equivalently, each edge e has a **weight** $w(e) \in \mathbb{Z}_2$.

We only care about the **weight** of each cycle, where $w(C) := \sum_{e \in E(C)} w(e)$ over $\mathbb{Z}_2$. So **re-signing** (adding 1 to each edge in a cut) gives an equivalent signed graph.



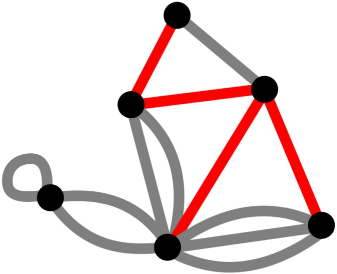
A **signed graph** is a graph with a specified set Σ of edges, called the **signature**. Equivalently, each edge e has a **weight** $w(e) \in \mathbb{Z}_2$.

We only care about the **weight** of each cycle, where $w(C) := \sum_{e \in E(C)} w(e)$ over $\mathbb{Z}_2$. So **re-signing** (adding 1 to each edge in a cut) gives an equivalent signed graph.



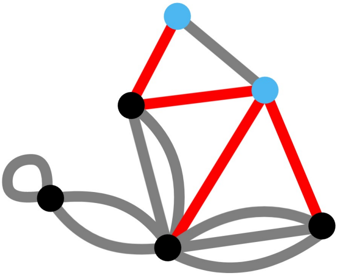
A **signed graph** is a graph with a specified set Σ of edges, called the **signature**. Equivalently, each edge e has a **weight** $w(e) \in \mathbb{Z}_2$.

We only care about the **weight** of each cycle, where $w(C) := \sum_{e \in E(C)} w(e)$ over $\mathbb{Z}_2$. So **re-signing** (adding 1 to each edge in a cut) gives an equivalent signed graph.



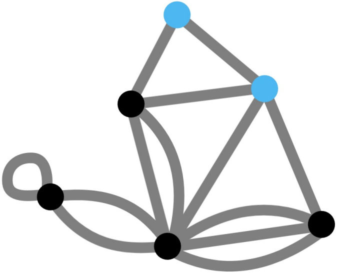
A **signed graph** is a graph with a specified set Σ of edges, called the **signature**. Equivalently, each edge e has a **weight** $w(e) \in \mathbb{Z}_2$.

We only care about the **weight** of each cycle, where $w(C) := \sum_{e \in E(C)} w(e)$ over $\mathbb{Z}_2$. So **re-signing** (adding 1 to each edge in a cut) gives an equivalent signed graph.



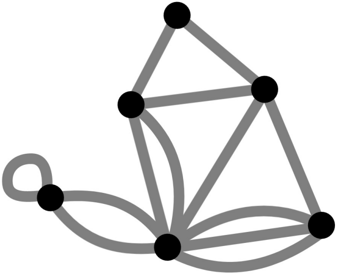
A **signed graph** is a graph with a specified set Σ of edges, called the **signature**. Equivalently, each edge e has a **weight** $w(e) \in \mathbb{Z}_2$.

We only care about the **weight** of each cycle, where $w(C) := \sum_{e \in E(C)} w(e)$ over $\mathbb{Z}_2$. So **re-signing** (adding 1 to each edge in a cut) gives an equivalent signed graph.



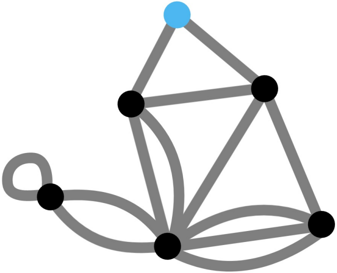
A **signed graph** is a graph with a specified set Σ of edges, called the **signature**. Equivalently, each edge e has a **weight** $w(e) \in \mathbb{Z}_2$.

We only care about the **weight** of each cycle, where $w(C) := \sum_{e \in E(C)} w(e)$ over $\mathbb{Z}_2$. So **re-signing** (adding 1 to each edge in a cut) gives an equivalent signed graph.



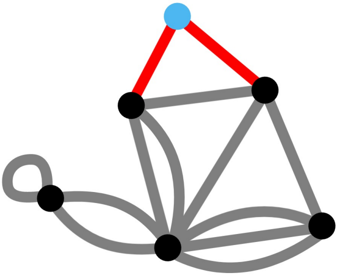
A **signed graph** is a graph with a specified set Σ of edges, called the **signature**. Equivalently, each edge e has a **weight** $w(e) \in \mathbb{Z}_2$.

We only care about the **weight** of each cycle, where $w(C) := \sum_{e \in E(C)} w(e)$ over $\mathbb{Z}_2$. So **re-signing** (adding 1 to each edge in a cut) gives an equivalent signed graph.



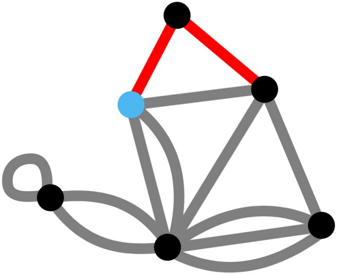
A **signed graph** is a graph with a specified set Σ of edges, called the **signature**. Equivalently, each edge e has a **weight** $w(e) \in \mathbb{Z}_2$.

We only care about the **weight** of each cycle, where $w(C) := \sum_{e \in E(C)} w(e)$ over $\mathbb{Z}_2$. So **re-signing** (adding 1 to each edge in a cut) gives an equivalent signed graph.



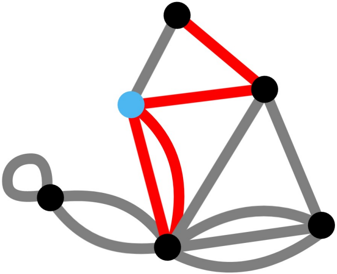
A **signed graph** is a graph with a specified set Σ of edges, called the **signature**. Equivalently, each edge e has a **weight** $w(e) \in \mathbb{Z}_2$.

We only care about the **weight** of each cycle, where $w(C) := \sum_{e \in E(C)} w(e)$ over $\mathbb{Z}_2$. So **re-signing** (adding 1 to each edge in a cut) gives an equivalent signed graph.



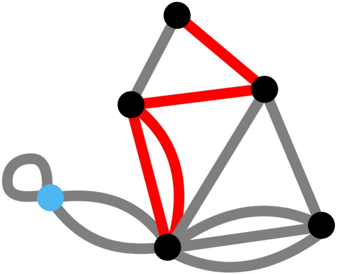
A **signed graph** is a graph with a specified set Σ of edges, called the **signature**. Equivalently, each edge e has a **weight** $w(e) \in \mathbb{Z}_2$.

We only care about the **weight** of each cycle, where $w(C) := \sum_{e \in E(C)} w(e)$ over $\mathbb{Z}_2$. So **re-signing** (adding 1 to each edge in a cut) gives an equivalent signed graph.



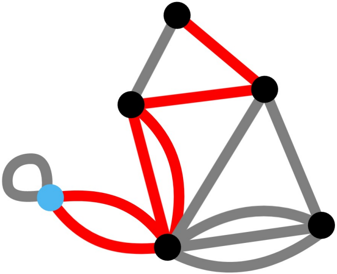
A **signed graph** is a graph with a specified set Σ of edges, called the **signature**. Equivalently, each edge e has a **weight** $w(e) \in \mathbb{Z}_2$.

We only care about the **weight** of each cycle, where $w(C) := \sum_{e \in E(C)} w(e)$ over $\mathbb{Z}_2$. So **re-signing** (adding 1 to each edge in a cut) gives an equivalent signed graph.



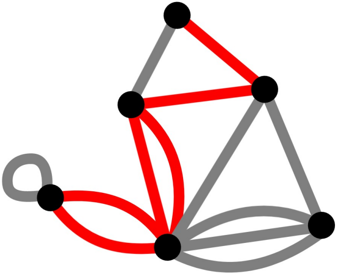
A **signed graph** is a graph with a specified set Σ of edges, called the **signature**. Equivalently, each edge e has a **weight** $w(e) \in \mathbb{Z}_2$.

We only care about the **weight** of each cycle, where $w(C) := \sum_{e \in E(C)} w(e)$ over $\mathbb{Z}_2$. So **re-signing** (adding 1 to each edge in a cut) gives an equivalent signed graph.



A **signed graph** is a graph with a specified set Σ of edges, called the **signature**. Equivalently, each edge e has a **weight** $w(e) \in \mathbb{Z}_2$.

We only care about the **weight** of each cycle, where $w(C) := \sum_{e \in E(C)} w(e)$ over $\mathbb{Z}_2$. So **re-signing** (adding 1 to each edge in a cut) gives an equivalent signed graph.



A **signed graph** is a graph with a specified set Σ of edges, called the **signature**. Equivalently, each edge e has a **weight** $w(e) \in \mathbb{Z}_2$.

We only care about the **weight** of each cycle, where $w(C) := \sum_{e \in E(C)} w(e)$ over $\mathbb{Z}_2$. So **re-signing** (adding 1 to each edge in a cut) gives an equivalent signed graph.

For **immersions** of signed graphs, after re-signing,

- each $e \in E(H)$ is sent to a trail of weight $w(e)$ in G.

Now **flooding immersions** behave much differently.

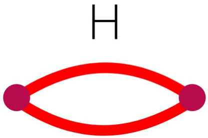
Graphs with k different signatures are called **$\mathbb{Z}_2^k$-labelled**; each edge has a weight $w(e) \in \mathbb{Z}_2^k$. We can **re-sign** on any $\gamma \in \mathbb{Z}_2^k$.

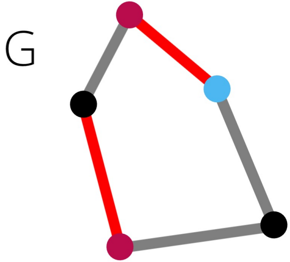
For **immersions** of signed graphs, after re-signing H,

- each $e \in E(H)$ is sent to a trail of weight $w(e)$ in G.

Now **flooding immersions** behave much differently.

Graphs with k different signatures are called **$\mathbb{Z}_2^k$-labelled**; each edge has a weight $w(e) \in \mathbb{Z}_2^k$. We can **re-sign** on any $\gamma \in \mathbb{Z}_2^k$.



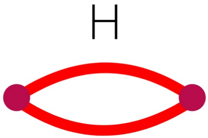


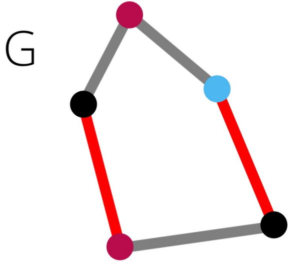
For **immersions** of signed graphs, after re-signing H,

- each $e \in E(H)$ is sent to a trail of weight $w(e)$ in G.

Now **flooding immersions** behave much differently.

Graphs with k different signatures are called **$\mathbb{Z}_2^k$-labelled**; each edge has a weight $w(e) \in \mathbb{Z}_2^k$. We can **re-sign** on any $\gamma \in \mathbb{Z}_2^k$.



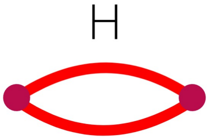


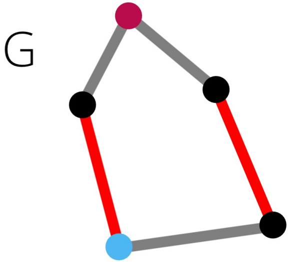
For **immersions** of signed graphs, after re-signing H,

- each $e \in E(H)$ is sent to a trail of weight $w(e)$ in G.

Now **flooding immersions** behave much differently.

Graphs with k different signatures are called **$\mathbb{Z}_2^k$-labelled**; each edge has a weight $w(e) \in \mathbb{Z}_2^k$. We can **re-sign** on any $\gamma \in \mathbb{Z}_2^k$.





For **immersions** of signed graphs, after re-signing H,

- each $e \in E(H)$ is sent to a trail of weight $w(e)$ in G.

Now **flooding immersions** behave much differently.

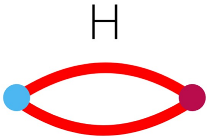
Graphs with k different signatures are called **$\mathbb{Z}_2^k$-labelled**; each edge has a weight $w(e) \in \mathbb{Z}_2^k$. We can **re-sign** on any $\gamma \in \mathbb{Z}_2^k$.

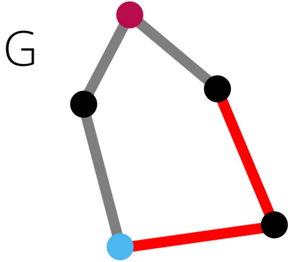
For **immersions** of signed graphs, after re-signing H,

- each $e \in E(H)$ is sent to a trail of weight $w(e)$ in G.

Now **flooding immersions** behave much differently.

Graphs with k different signatures are called **$\mathbb{Z}_2^k$-labelled**; each edge has a weight $w(e) \in \mathbb{Z}_2^k$. We can **re-sign** on any $\gamma \in \mathbb{Z}_2^k$.





For **immersions** of signed graphs, after re-signing H,

- each $e \in E(H)$ is sent to a trail of weight $w(e)$ in G.

Now **flooding immersions** behave much differently.

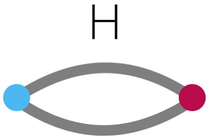
Graphs with k different signatures are called **$\mathbb{Z}_2^k$-labelled**; each edge has a weight $w(e) \in \mathbb{Z}_2^k$. We can **re-sign** on any $\gamma \in \mathbb{Z}_2^k$.

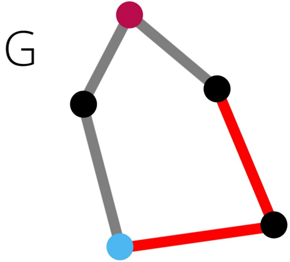
For **immersions** of signed graphs, after re-signing H,

- each $e \in E(H)$ is sent to a trail of weight $w(e)$ in G.

Now **flooding immersions** behave much differently.

Graphs with k different signatures are called **$\mathbb{Z}_2^k$-labelled**; each edge has a weight $w(e) \in \mathbb{Z}_2^k$. We can **re-sign** on any $\gamma \in \mathbb{Z}_2^k$.



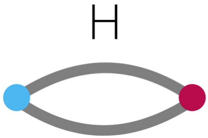


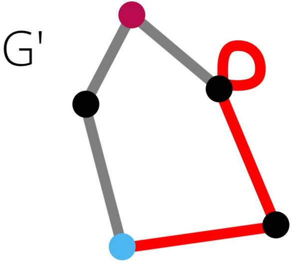
For **immersions** of signed graphs, after re-signing H,

- each $e \in E(H)$ is sent to a trail of weight $w(e)$ in G.

Now **flooding immersions** behave much differently.

Graphs with k different signatures are called **$\mathbb{Z}_2^k$-labelled**; each edge has a weight $w(e) \in \mathbb{Z}_2^k$. We can **re-sign** on any $\gamma \in \mathbb{Z}_2^k$.





For **immersions** of signed graphs, after re-signing H,

- each $e \in E(H)$ is sent to a trail of weight $w(e)$ in G.

Now **flooding immersions** behave much differently.

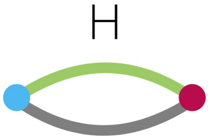
Graphs with k different signatures are called **$\mathbb{Z}_2^k$-labelled**; each edge has a weight $w(e) \in \mathbb{Z}_2^k$. We can **re-sign** on any $\gamma \in \mathbb{Z}_2^k$.

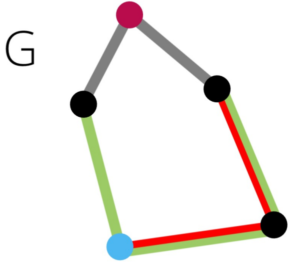
For **immersions** of signed graphs, after re-signing H,

- each $e \in E(H)$ is sent to a trail of weight $w(e)$ in G.

Now **flooding immersions** behave much differently.

Graphs with k different signatures are called **$\mathbb{Z}_2^k$-labelled**; each edge has a weight $w(e) \in \mathbb{Z}_2^k$. We can **re-sign** on any $\gamma \in \mathbb{Z}_2^k$.



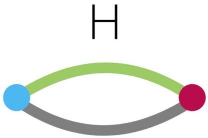


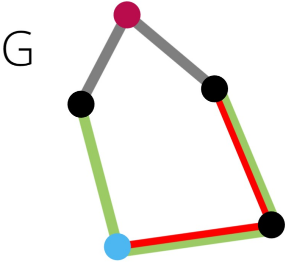
For **immersions** of signed graphs, after re-signing H,

- each $e \in E(H)$ is sent to a trail of weight $w(e)$ in G.

Now **flooding immersions** behave much differently.

Graphs with k different signatures are called **$\mathbb{Z}_2^k$-labelled**; each edge has a weight $w(e) \in \mathbb{Z}_2^k$. We can **re-sign** on any $\gamma \in \mathbb{Z}_2^k$.





For **immersions** of signed graphs, after re-signing H,

- each $e \in E(H)$ is sent to a trail of weight $w(e)$ in G.

Now **flooding immersions** behave much differently.

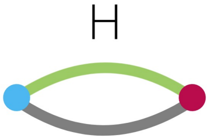
Graphs with k different signatures are called $\mathbb{Z}_2^k$**-labelled**; each edge has a weight $w(e) \in \mathbb{Z}_2^k$. We can **re-sign** on any $\gamma \in \mathbb{Z}_2^k$.

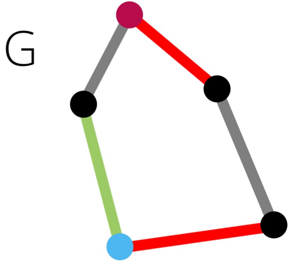
For **immersions** of signed graphs, after re-signing H,

- each $e \in E(H)$ is sent to a trail of weight $w(e)$ in G.

Now **flooding immersions** behave much differently.

Graphs with k different signatures are called **$\mathbb{Z}_2^k$-labelled**; each edge has a weight $w(e) \in \mathbb{Z}_2^k$. We can **re-sign** on any $\gamma \in \mathbb{Z}_2^k$.





For **immersions** of signed graphs, after re-signing H,

- each $e \in E(H)$ is sent to a trail of weight $w(e)$ in G.

Now **flooding immersions** behave much differently.

Graphs with k different signatures are called **$\mathbb{Z}_2^k$-labelled**; each edge has a weight $w(e) \in \mathbb{Z}_2^k$. We can **re-sign** on any $\gamma \in \mathbb{Z}_2^k$.

So we are interested in **flooding immersions** of $\mathbb{Z}_2^k$-labelled Eulerian graphs...

...in order to describe the structure of graphs without H as a **vertex-minor**.

The connection is through **circle graphs**.

So we are interested in **flooding immersions** of $\mathbb{Z}_2^k$-labelled Eulerian graphs...

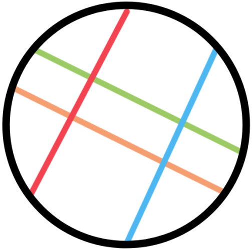
...in order to describe the structure of graphs without H as a **vertex-minor**.

The connection is through **circle graphs**.

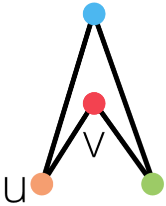
So we are interested in **flooding immersions** of $\mathbb{Z}_2^k$-labelled Eulerian graphs...

...in order to describe the structure of graphs without H as a **vertex-minor**.

The connection is through **circle graphs**.



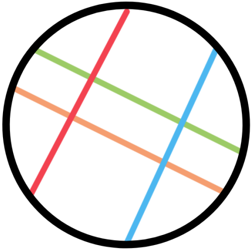
chord diagram



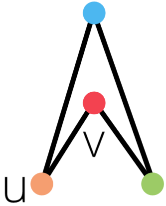
circle graph G

A **circle graph** is the intersection graph of chords on a circle.
Circle graphs are closed under local complementation.

View the **chord diagram** as a 3-regular graph and contract the chords to get the **tour graph**. It is invariant under local complementation. Here is how we delete v.



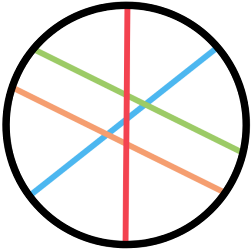
chord diagram



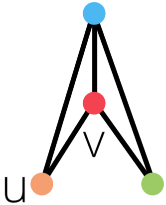
circle graph G

A **circle graph** is the intersection graph of chords on a circle. Circle graphs are closed under local complementation.

View the **chord diagram** as a 3-regular graph and contract the chords to get the **tour graph**. It is invariant under local complementation. Here is how we delete v.



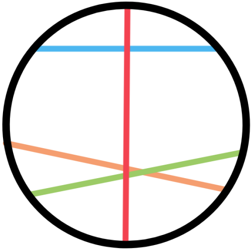
chord diagram



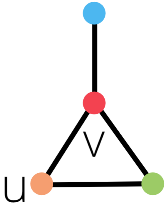
circle graph $G * u$

A **circle graph** is the intersection graph of chords on a circle. Circle graphs are closed under local complementation.

View the **chord diagram** as a 3-regular graph and contract the chords to get the **tour graph**. It is invariant under local complementation. Here is how we delete v.



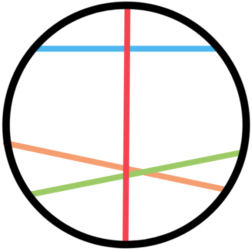
chord diagram



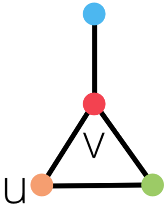
circle graph $G * u * v$

A **circle graph** is the intersection graph of chords on a circle. Circle graphs are closed under local complementation.

View the **chord diagram** as a 3-regular graph and contract the chords to get the **tour graph**. It is invariant under local complementation. Here is how we delete v.



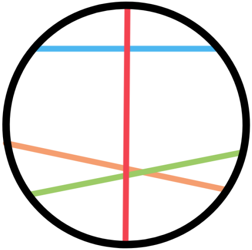
chord diagram



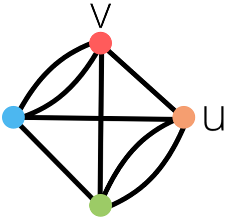
circle graph $G * u * v$

A **circle graph** is the intersection graph of chords on a circle. Circle graphs are closed under local complementation.

View the **chord diagram** as a 3-regular graph and contract the chords to get the **tour graph**. It is invariant under local complementation. Here is how we delete v.



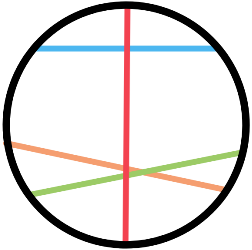
chord diagram



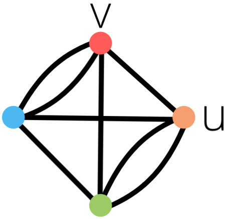
tour graph

A **circle graph** is the intersection graph of chords on a circle. Circle graphs are closed under local complementation.

View the **chord diagram** as a 3-regular graph and contract the chords to get the **tour graph**. It is invariant under local complementation. Here is how we delete v.



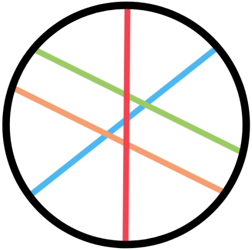
chord diagram



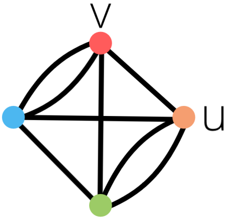
tour graph

A **circle graph** is the intersection graph of chords on a circle. Circle graphs are closed under local complementation.

View the **chord diagram** as a 3-regular graph and contract the chords to get the **tour graph**. It is invariant under local complementation. Here is how we delete v.



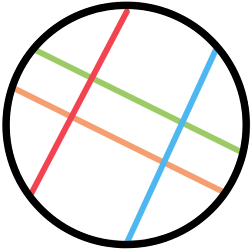
chord diagram



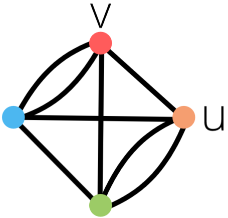
tour graph

A **circle graph** is the intersection graph of chords on a circle. Circle graphs are closed under local complementation.

View the **chord diagram** as a 3-regular graph and contract the chords to get the **tour graph**. It is invariant under local complementation. Here is how we delete v.



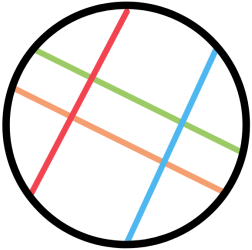
chord diagram



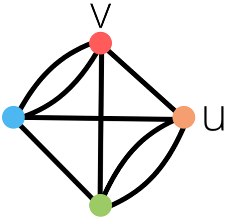
tour graph

A **circle graph** is the intersection graph of chords on a circle. Circle graphs are closed under local complementation.

View the **chord diagram** as a 3-regular graph and contract the chords to get the **tour graph**. It is invariant under local complementation. Here is how we delete *v*.



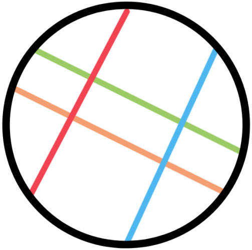
chord diagram



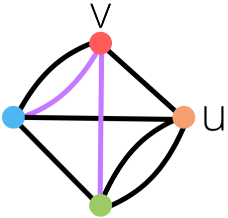
tour graph

A **circle graph** is the intersection graph of chords on a circle. Circle graphs are closed under local complementation.

View the **chord diagram** as a 3-regular graph and contract the chords to get the **tour graph**. It is invariant under local complementation. Here is how we delete v.



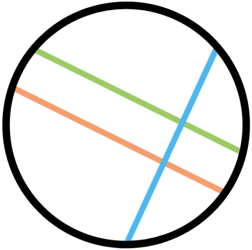
chord diagram



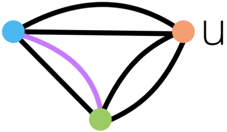
tour graph

A **circle graph** is the intersection graph of chords on a circle. Circle graphs are closed under local complementation.

View the **chord diagram** as a 3-regular graph and contract the chords to get the **tour graph**. It is invariant under local complementation. Here is how we delete v.



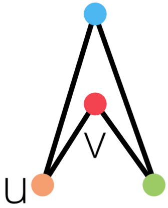
chord diagram



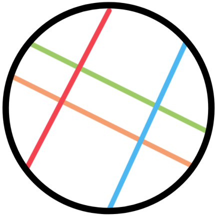
tour graph

A **circle graph** is the intersection graph of chords on a circle. Circle graphs are closed under local complementation.

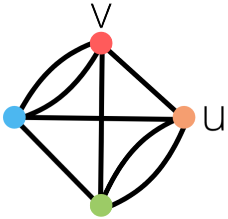
View the **chord diagram** as a 3-regular graph and contract the chords to get the **tour graph**. It is invariant under local complementation. Here is how we delete v.



circle graph



chord diagram

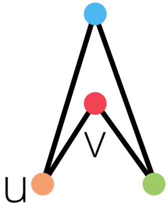


tour graph

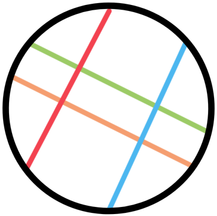
Theorem (Kotzig 77 and Bouchet 94)

If H and G are prime circle graphs, then

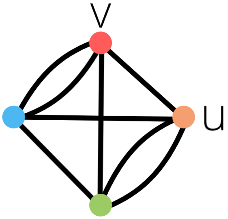
- *their tour graphs $T(H)$ and $T(G)$ are unique and*
- *H is a* **vertex-minor** *of G $\iff$ $T(H)$* **floods** *$T(G)$.*



circle graph



chord diagram

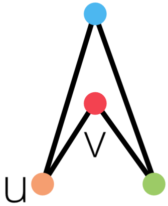


tour graph

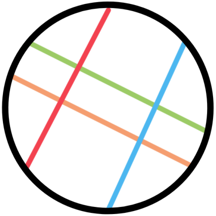
Theorem (Kotzig 77 and Bouchet 94)

If H and G are prime circle graphs, then

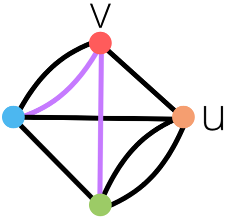
- *their tour graphs $T(H)$ and $T(G)$ are unique and*
- *H is a* **vertex-minor** *of G $\iff$ $T(H)$* **floods** *$T(G)$.*



circle graph



chord diagram

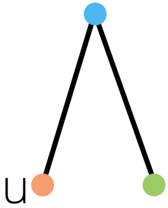


tour graph

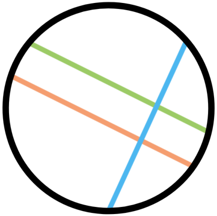
Theorem (Kotzig 77 and Bouchet 94)

If H and G are prime circle graphs, then

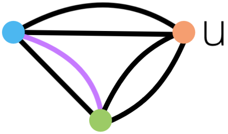
- *their tour graphs $T(H)$ and $T(G)$ are unique and*
- *H is a* **vertex-minor** *of G $\iff$ $T(H)$* **floods** *$T(G)$.*



circle graph



chord diagram



tour graph

Theorem (Kotzig 77 and Bouchet 94)

If H and G are prime circle graphs, then

- *their tour graphs $T(H)$ and $T(G)$ are unique and*
- *H is a* **vertex-minor** *of G $\iff$ $T(H)$* **floods** *$T(G)$.*

Theorem (with O-joung Kwon)

A class of graphs has unbounded rank-width if and only if it contains all **circle graphs** *as* **vertex-minors***.*

Theorem (with O-joung Kwon)

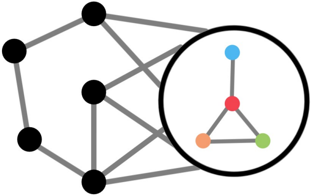
A class of graphs has unbounded rank-width if and only if it contains all **circle graphs** *as* **vertex-minors***.*

Suppose G does not have H as a vertex-minor.

Theorem (with O-joung Kwon)

A class of graphs has unbounded rank-width if and only if it contains all **circle graphs** *as* **vertex-minors***.*

Suppose G does not have H as a vertex-minor.

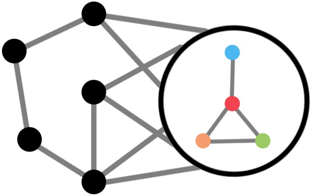


WMA that after local complementation, our favorite circle graph is an induced subgraph of G.

Theorem (with O-joung Kwon)

A class of graphs has unbounded rank-width if and only if it contains all **circle graphs** *as* **vertex-minors***.*

Suppose G does not have H as a vertex-minor.

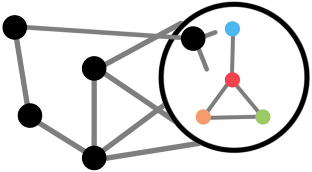


WMA that after local complementation, our favorite circle graph is an induced subgraph of G. Throw vertices into the circle graph.

Theorem (with O-joung Kwon)

A class of graphs has unbounded rank-width if and only if it contains all **circle graphs** *as* **vertex-minors**.

Suppose G does not have H as a vertex-minor.

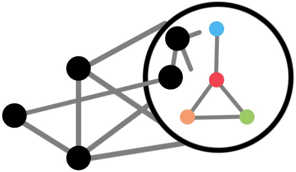


WMA that after local complementation, our favorite circle graph is an induced subgraph of G. Throw vertices into the circle graph.

Theorem (with O-joung Kwon)

A class of graphs has unbounded rank-width if and only if it contains all **circle graphs** *as* **vertex-minors***.*

Suppose G does not have H as a vertex-minor.

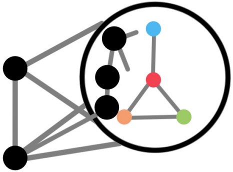


WMA that after local complementation, our favorite circle graph is an induced subgraph of G. Throw vertices into the circle graph.

Theorem (with O-joung Kwon)

A class of graphs has unbounded rank-width if and only if it contains all **circle graphs** *as* **vertex-minors***.*

Suppose G does not have H as a vertex-minor.

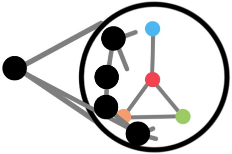


WMA that after local complementation, our favorite circle graph is an induced subgraph of G. Throw vertices into the circle graph.

Theorem (with O-joung Kwon)

A class of graphs has unbounded rank-width if and only if it contains all **circle graphs** *as* **vertex-minors***.*

Suppose G does not have H as a vertex-minor.

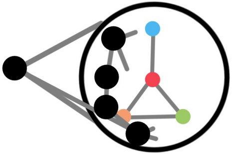


WMA that after local complementation, our favorite circle graph is an induced subgraph of G. Throw vertices into the circle graph.

Theorem (with O-joung Kwon)

A class of graphs has unbounded rank-width if and only if it contains all **circle graphs** *as* **vertex-minors***.*

Suppose G does not have H as a vertex-minor.



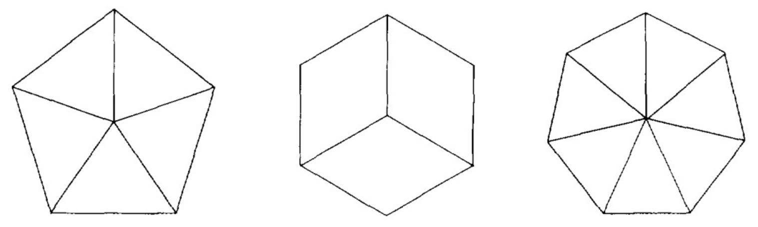
WMA that after local complementation, our favorite circle graph is an induced subgraph of G. Throw vertices into the circle graph. The remaining vertex gives a **signature** on the **tour graph**.

Theorem (Bouchet 94)

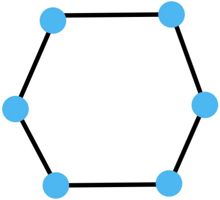
A graph is a circle graph

$$\iff$$

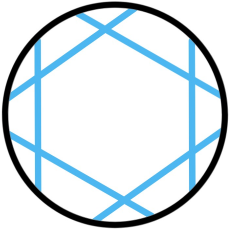
it has none of



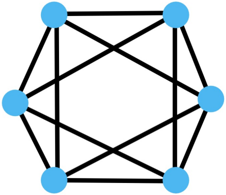
as a vertex-minor.



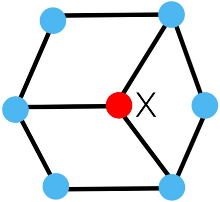
circle graph



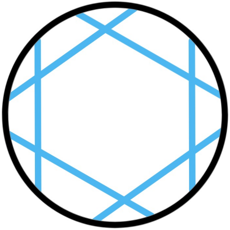
chord diagram



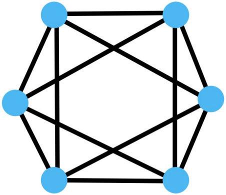
tour graph



circle graph + x

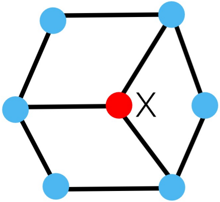


chord diagram

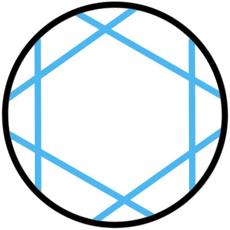


tour graph

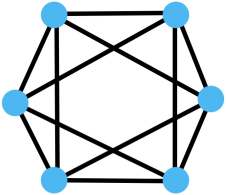
Adding x will give a signature Σ in the **tour graph**.



circle graph + x

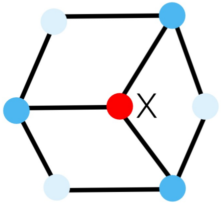


chord diagram

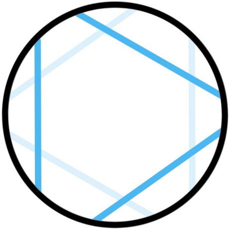


tour graph

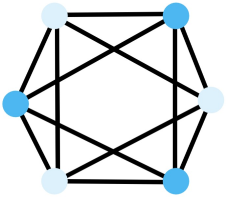
Adding x will give a signature Σ in the **tour graph**. Consider the neighbourhood of x.



circle graph + x

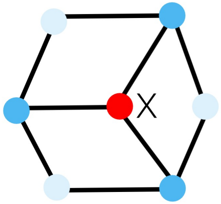


chord diagram

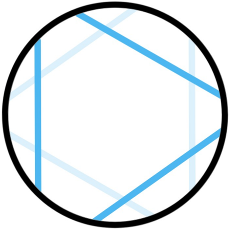


tour graph

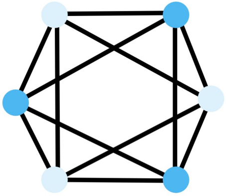
Adding x will give a signature Σ in the **tour graph**. Consider the neighbourhood of x.



circle graph + x

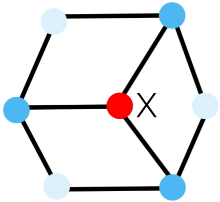


chord diagram

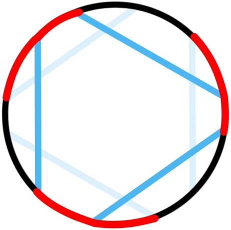


tour graph

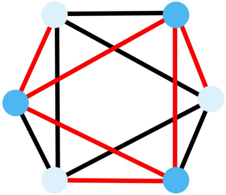
Adding x will give a signature Σ in the **tour graph**. Consider the neighbourhood of *x*. For each neighbour, choose an end of its chord and add 1 to both “incident arcs”.



circle graph + x

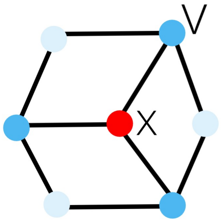


chord diagram

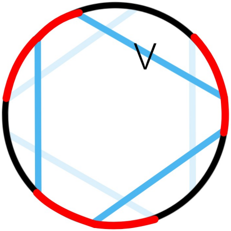


tour graph

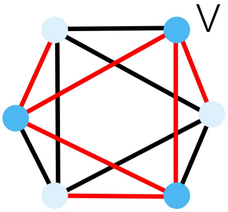
Adding x will give a signature Σ in the **tour graph**. Consider the neighbourhood of x. For each neighbour, choose an end of its chord and add 1 to both “incident arcs”.



circle graph + x



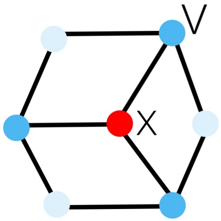
chord diagram



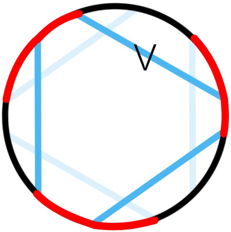
tour graph

Adding x will give a signature Σ in the **tour graph**. Consider the neighbourhood of x. For each neighbour, choose an end of its chord and add 1 to both "incident arcs".

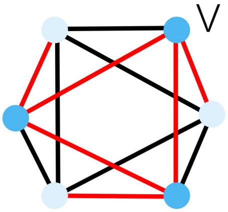
Then v is a neighbour $\iff$ the chord of v "splits the circle into two odd parts".



circle graph + x



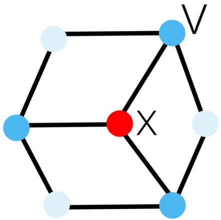
chord diagram



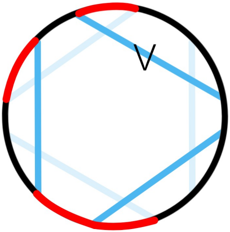
tour graph

Adding x will give a signature Σ in the **tour graph**. Consider the neighbourhood of x. For each neighbour, choose an end of its chord and add 1 to both "incident arcs".

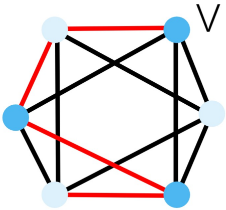
Then v is a neighbour $\iff$ the chord of v "splits the circle into two odd parts". Re-signing at v is OK.



circle graph + x



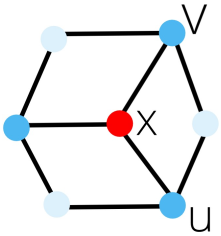
chord diagram



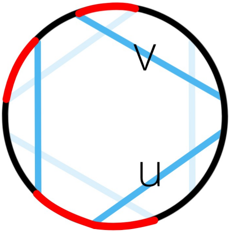
tour graph

Adding x will give a signature Σ in the **tour graph**. Consider the neighbourhood of x. For each neighbour, choose an end of its chord and add 1 to both “incident arcs”.

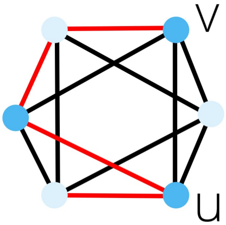
Then v is a neighbour $\iff$ the chord of v “splits the circle into two odd parts”. Re-signing at v is OK.



circle graph + x



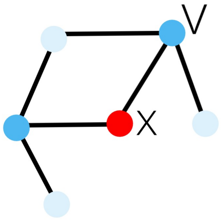
chord diagram



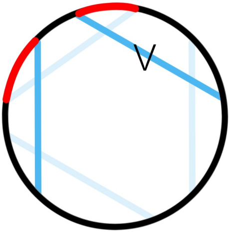
tour graph

Adding x will give a signature Σ in the **tour graph**. Consider the neighbourhood of x. For each neighbour, choose an end of its chord and add 1 to both "incident arcs".

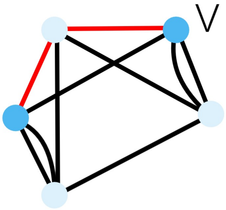
Then v is a neighbour $\iff$ the chord of v "splits the circle into two odd parts". Re-signing at v is OK. Deleting u is OK.



circle graph + x



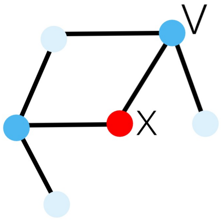
chord diagram



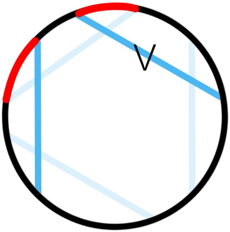
tour graph

Adding x will give a signature Σ in the **tour graph**. Consider the neighbourhood of x. For each neighbour, choose an end of its chord and add 1 to both "incident arcs".

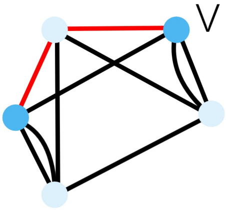
Then v is a neighbour $\iff$ the chord of v "splits the circle into two odd parts". Re-signing at v is OK. Deleting u is OK.



circle graph + x



chord diagram

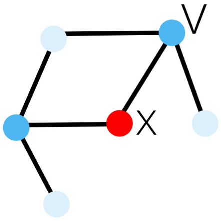


tour graph

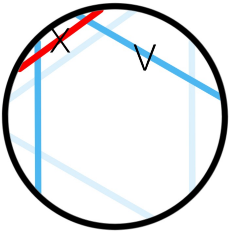
Adding x will give a signature Σ in the **tour graph**. Consider the neighbourhood of x. For each neighbour, choose an end of its chord and add 1 to both "incident arcs".

Then v is a neighbour $\iff$ the chord of v "splits the circle into two odd parts". Re-signing at v is OK. Deleting u is OK.

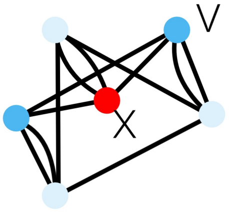
Can add x as a chord $\iff$ can re-sign s.t. $|\Sigma| \leq 2$.



circle graph



chord diagram



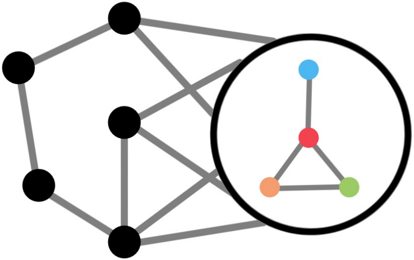
tour graph

Adding x will give a signature Σ in the **tour graph**. Consider the neighbourhood of x. For each neighbour, choose an end of its chord and add 1 to both "incident arcs".

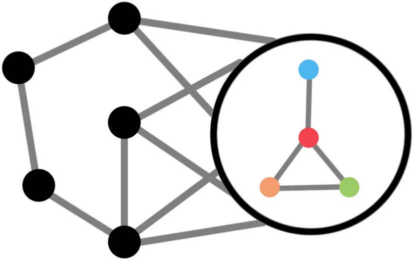
Then v is a neighbour $\iff$ the chord of v "splits the circle into two odd parts". Re-signing at v is OK. Deleting u is OK.

Can add x as a chord $\iff$ can re-sign s.t. $|\Sigma| \leq 2$.

What if no more vertices can be added to the circle graph?

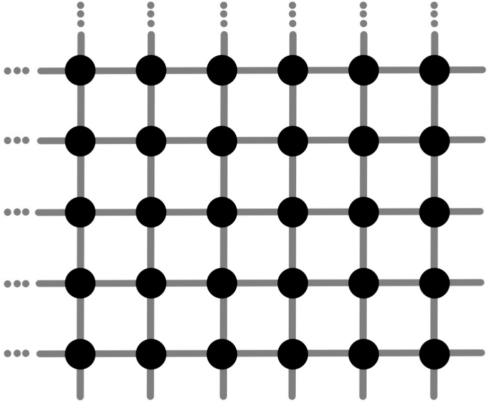


What if no more vertices can be added to the circle graph?



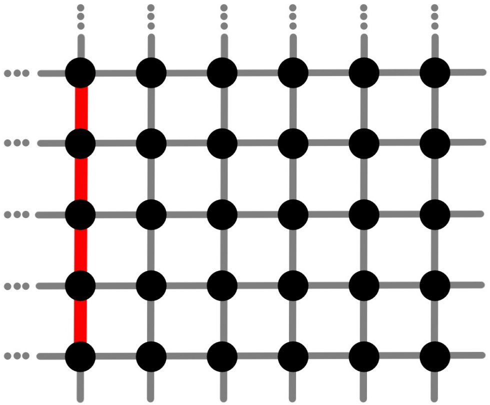
Then the **tour graph** is $\mathbb{Z}_2^k$-**labelled**.

Here is an example of a "win"...



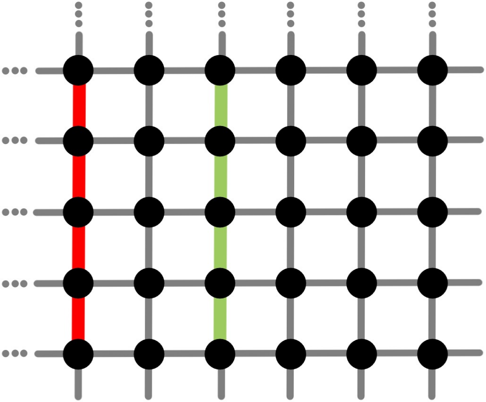
A toroidal grid with signatures Σ_1, Σ_2, Σ_3 of size 4, "far apart".

Here is an example of a “win”...



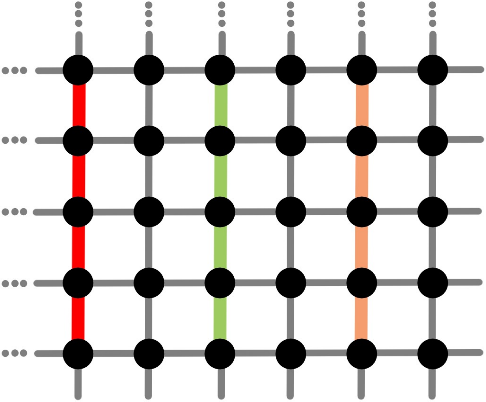
A toroidal grid with signatures Σ_1, Σ_2, Σ_3
of size 4, “far apart”.

Here is an example of a “win”...



A toroidal grid with signatures Σ_1, Σ_2, Σ_3 of size 4, “far apart”.

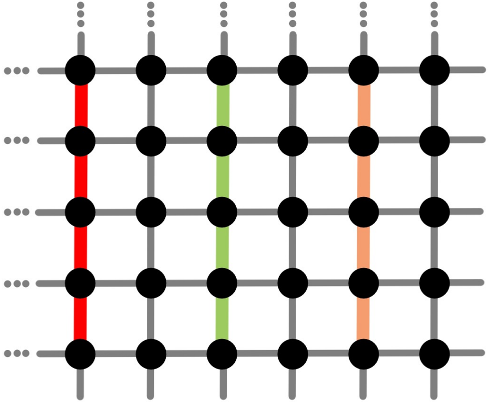
Here is an example of a "win"...



A toroidal grid with signatures Σ_1, Σ_2, Σ_3

of size 4, "far apart".

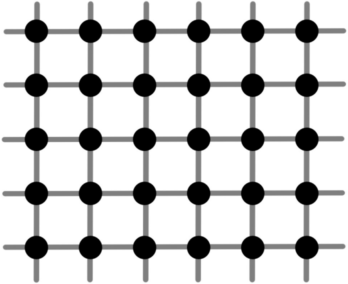
Here is an example of a “win”...



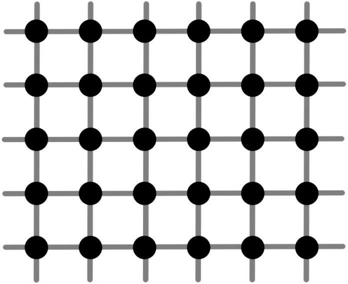
A toroidal grid with signatures Σ_1, Σ_2, Σ_3 of size 4, “far apart”.

Suppose we have a grid subgraph

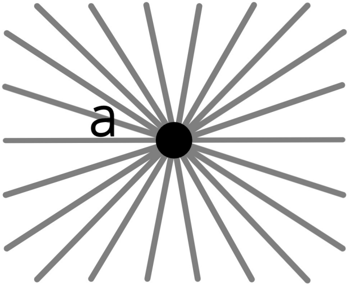
and we identify its vertices to a new vertex a.



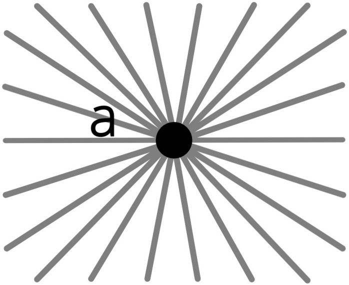
Suppose we have a grid subgraph
and we identify its vertices to a new vertex a.



Suppose we have a grid subgraph
and we identify its vertices to a new vertex *a*.

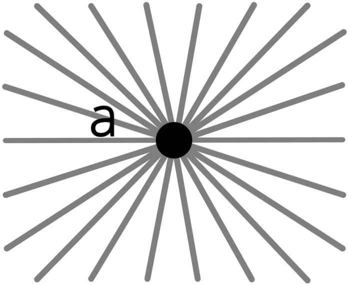


Suppose we have a grid subgraph
and we identify its vertices to a new vertex *a*.



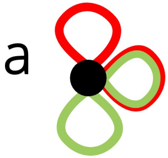
Need a "non-zero *A*-paths" type result...

Suppose we have a grid subgraph
and we identify its vertices to a new vertex a.



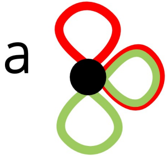
What is the maximum t s.t. G is **flooded** by
a 1-vertex graph with t non-zero loops at a?

Suppose we have a grid subgraph
and we identify its vertices to a new vertex a.



What is the maximum t s.t. G is **flooded** by
a 1-vertex graph with t non-zero loops at a?

Suppose we have a grid subgraph
and we identify its vertices to a new vertex a.



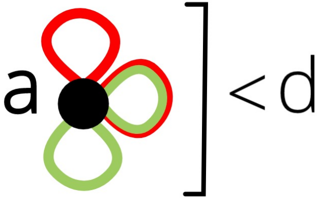
What is the maximum t s.t. G is **flooded** by a 1-vertex graph with t non-zero loops at a?

Have precise min-max theorem for $\mathbb{Z}_2^k$-**labelled** graphs.

Corollary

If G is $\mathbb{Z}_2^k$-labelled, Eulerian, and $2d$-**edge-connected** *for an integer $d \geq 2$, and $a \in V(G)$ with* $\max t < d$, *then there exist $S \subseteq V(G) \setminus \{a\}$, $\gamma \in \mathbb{Z}_2^k$, and a re-signing s.t.*

1. *every non-zero edge is incident to a vertex in S and has weight γ, and*
2. $|\delta(S)| = 2d$ *and* $w(E(G)) \neq d\gamma$.



Corollary

*If G is $\mathbb{Z}_2^k$-labelled, Eulerian, and $2d$-**edge-connected** for an integer $d \geq 2$, and $a \in V(G)$ with $\max t < d$, then there exist $S \subseteq V(G) \setminus \{a\}$, $\gamma \in \mathbb{Z}_2^k$, and a re-signing s.t.*

1. *every non-zero edge is incident to a vertex in S and has weight γ, and*
2. *$|\delta(S)| = 2d$ and $w(E(G)) \neq d\gamma$.*

Corollary

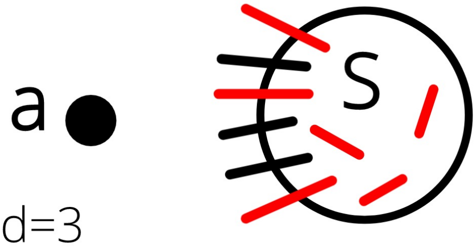
If G is $\mathbb{Z}_2^k$-labelled, Eulerian, and $2d$-**edge-connected** *for an integer $d \geq 2$, and $a \in V(G)$ with* $\max t < d$, *then there exist $S \subseteq V(G) \setminus \{a\}$, $\gamma \in \mathbb{Z}_2^k$, and a re-signing s.t.*

1. *every non-zero edge is incident to a vertex in S and has weight γ, and*
2. *$|\delta(S)| = 2d$ and $w(E(G)) \neq d\gamma$.*

Corollary

If G is $\mathbb{Z}_2^k$-labelled, Eulerian, and **$2d$-edge-connected** *for an integer $d \geq 2$, and $a \in V(G)$ with $\max t < d$, then there exist $S \subseteq V(G) \setminus \{a\}$, $\gamma \in \mathbb{Z}_2^k$, and a re-signing s.t.*

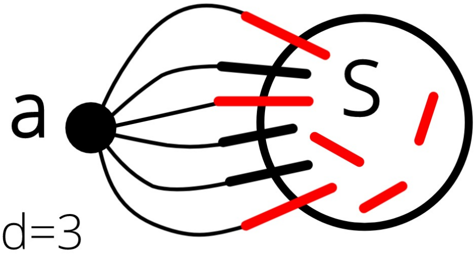
1. *every non-zero edge is incident to a vertex in S and has weight γ, and*
2. $|\delta(S)| = 2d$ *and* $w(E(G)) \neq d\gamma$.



Corollary

If G is $\mathbb{Z}_2^k$-labelled, Eulerian, and $2d$-**edge-connected** *for an integer $d \geq 2$, and $a \in V(G)$ with* $\max t < d$, *then there exist $S \subseteq V(G) \setminus \{a\}$, $\gamma \in \mathbb{Z}_2^k$, and a re-signing s.t.*

1. *every non-zero edge is incident to a vertex in S and has weight γ, and*
2. $|\delta(S)| = 2d$ *and* $w(E(G)) \neq d\gamma$.



Corollary

If G is $\mathbb{Z}_2^k$-labelled, Eulerian, and $2d$-**edge-connected** *for an integer $d \geq 2$, and $a \in V(G)$ with $\max t < d$, then there exist $S \subseteq V(G) \setminus \{a\}$, $\gamma \in \mathbb{Z}_2^k$, and a re-signing s.t.*

1. *every non-zero edge is incident to a vertex in S and has weight γ, and*
2. $|\delta(S)| = 2d$ *and* $w(E(G)) \neq d\gamma$.

Can we describe the structure of graphs with no **vertex-minor** isomorphic to H?

Can we describe the structure of graphs with no **vertex-minor** isomorphic to H?

Conjecture (Kim-Kwon-Oum-Sivaraman 20)**:**
They have chromatic number $\leq$ **polynomial**$_H$(clique number).

- See (Davies-M 20) and (Bonamy-Pilipczuk 20).

Can we describe the structure of graphs with no **vertex-minor** isomorphic to H?

Conjecture (Kim-Kwon-Oum-Sivaraman 20)**:**
They have chromatic number $\leq$ **polynomial**$_H$(clique number).

- See (Davies-M 20) and (Bonamy-Pilipczuk 20).

Conjecture:
Their clique number can be computed in polynomial-time.

- See (Courcelle-Makowsky-Rotics 99) and (Gavril 73).

Can we describe the structure of graphs with no **vertex-minor** isomorphic to H?

Conjecture (Kim-Kwon-Oum-Sivaraman 20)**:**
They have chromatic number $\leq$ **polynomial**$_H$(clique number).

- See (Davies-M 20) and (Bonamy-Pilipczuk 20).

Conjecture:
Their clique number can be computed in polynomial-time.

- See (Courcelle-Makowsky-Rotics 99) and (Gavril 73).

Conjecture: Graphs are well-quasi-ordered by vertex-minors.

- See (Oum 08).

Thank you!