A proof of the Multiplicative 1-2-3 Conjecture

Julien Bensmail[1], Hervé Hocquard[2], Dimitri Lajou[2], Éric Sopena[2]

1: I3S/INRIA – Université Côte d'Azur, France
2: LaBRI – Université de Bordeaux, France

Séminaire G&O, LaBRI
September 17th, 2021

Introduction

The 1-2-3 Conjecture, in few words

"Given a graph, can we assign 1, 2, 3 to its edges, so that
no two adjacent vertices are incident to the same sum of labels?"

The 1-2-3 Conjecture, in few words

"Given a graph, can we assign 1, 2, 3 to its edges, so that no two adjacent vertices are incident to the same sum of labels?"

Edge weights and vertex colours

Michał Karoński and Tomasz Łuczak

Faculty of Mathematics and Computer Science, Adam Mickiewicz University, Poznań, Poland

E-mail: karonski@amu.edu.pl and tomasz@amu.edu.pl

and

Andrew Thomason

DPMMS, Centre for Mathematical Sciences, Wilberforce Road, Cambridge CB3 0WB, England

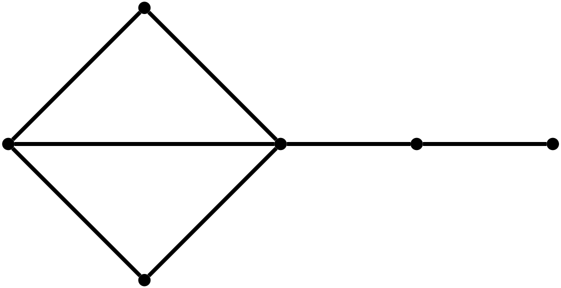
E-mail: a.g.thomason@dpmms.cam.ac.uk

Received 24th September 2002

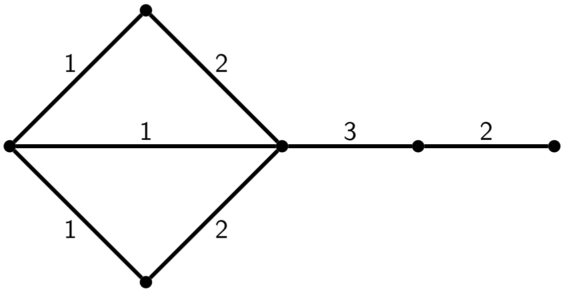
Can the edges of any non-trivial graph be assigned weights from $\{1, 2, 3\}$ so that adjacent vertices have different sums of incident edge weights?

We give a positive answer when the graph is 3-colourable, or when a finite number of real weights is allowed.

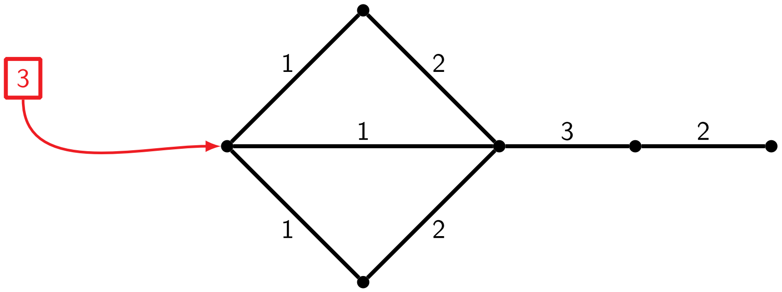
Sample example



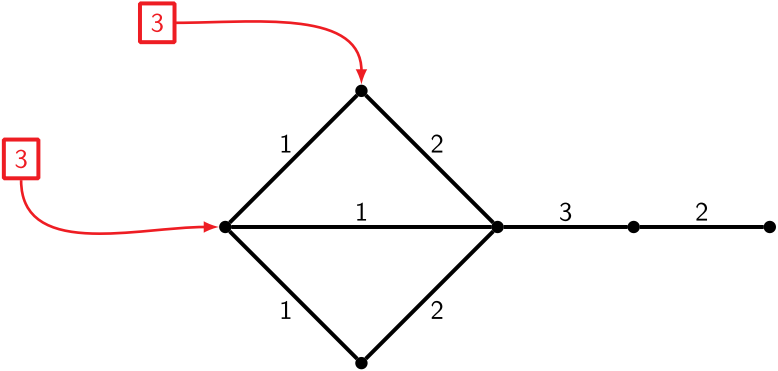
Sample example



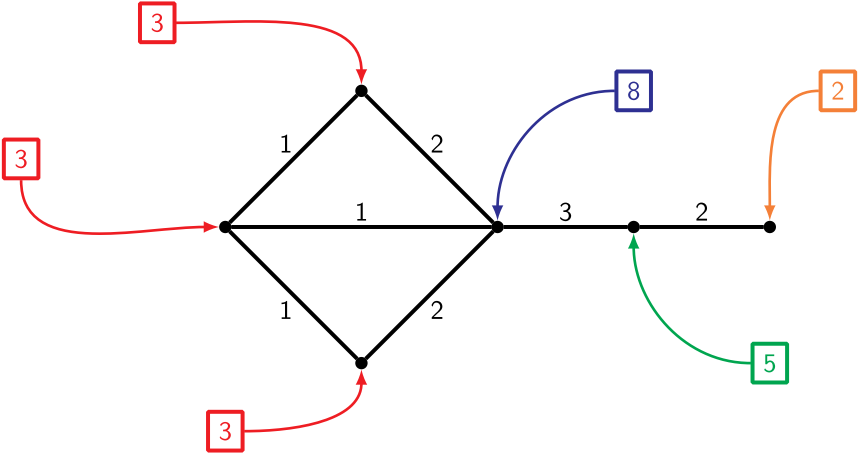
Sample example



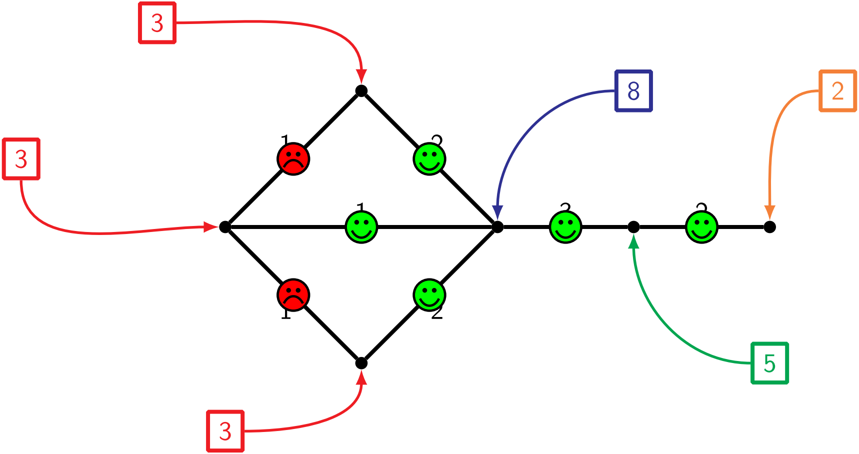
Sample example



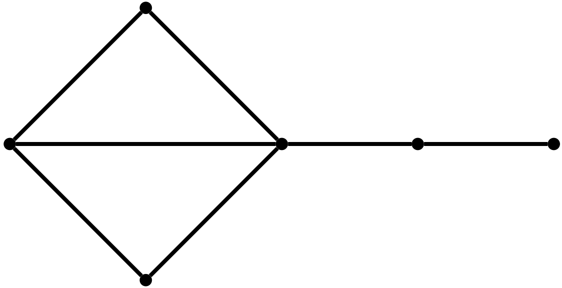
Sample example



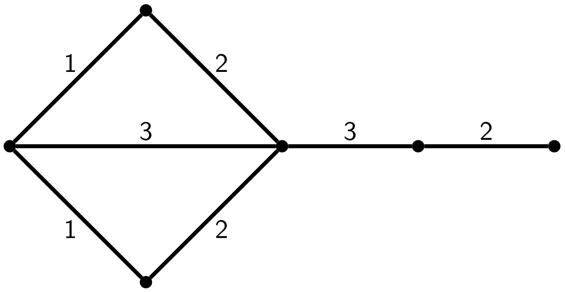
Sample example



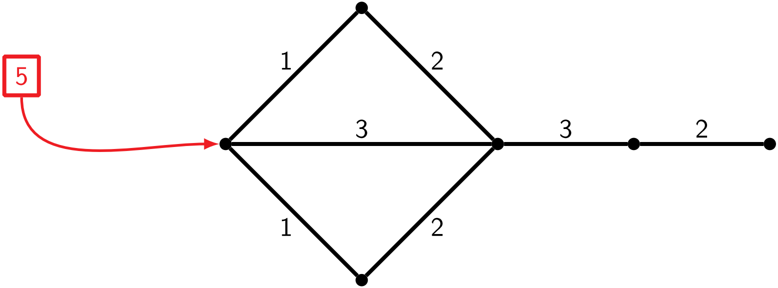
Sample example, 2nd try



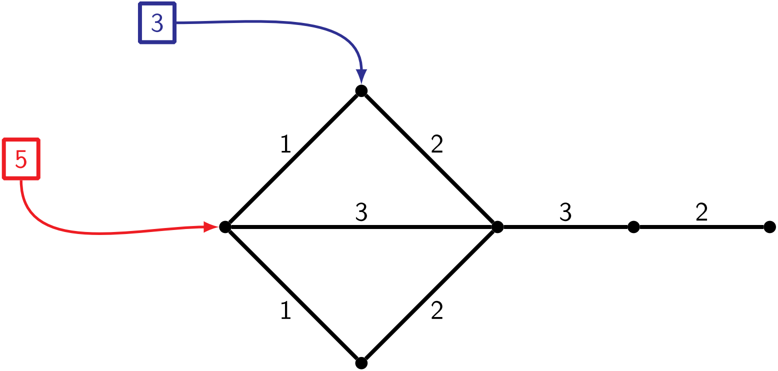
Sample example, 2nd try



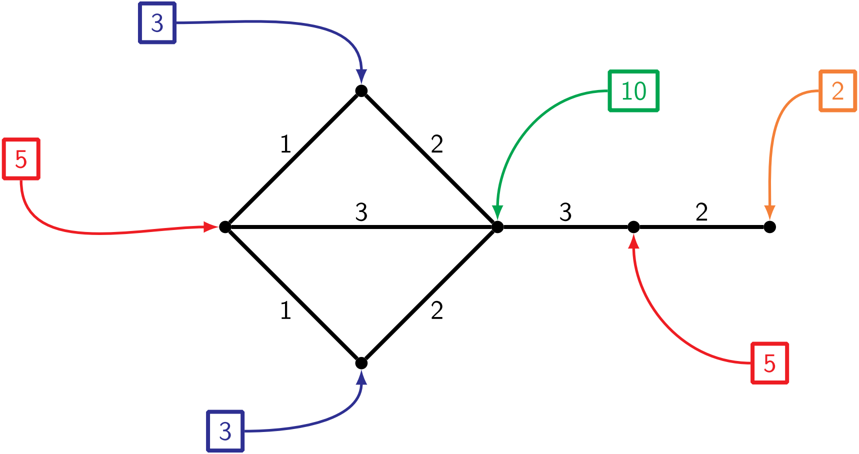
Sample example, 2nd try



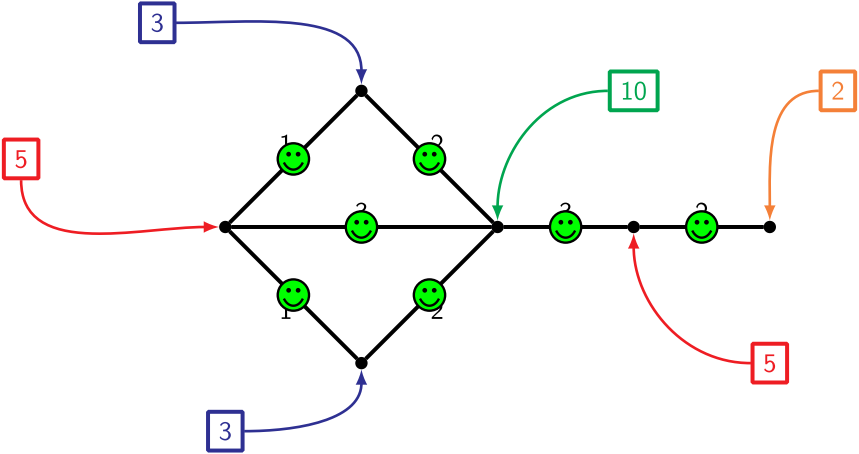
Sample example, 2nd try



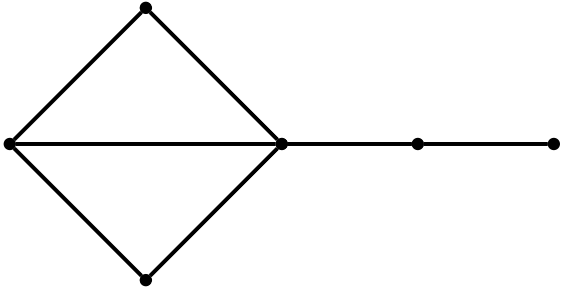
Sample example, 2nd try



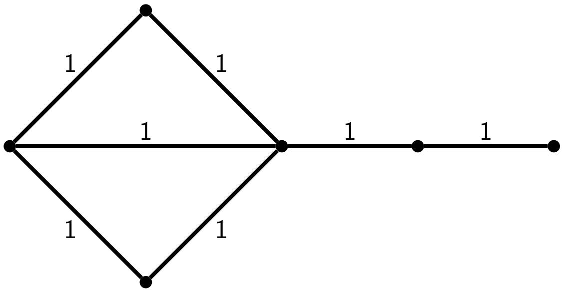
Sample example, 2nd try



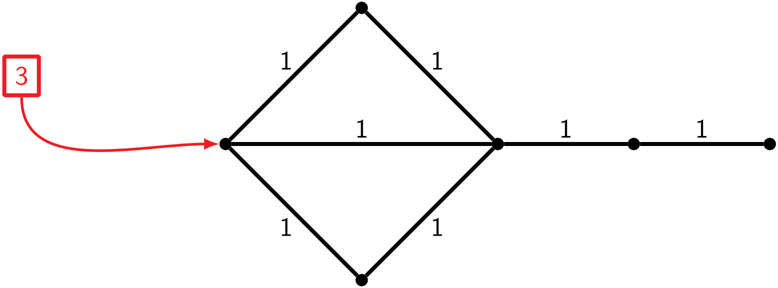
Sample example, 2nd try (again)



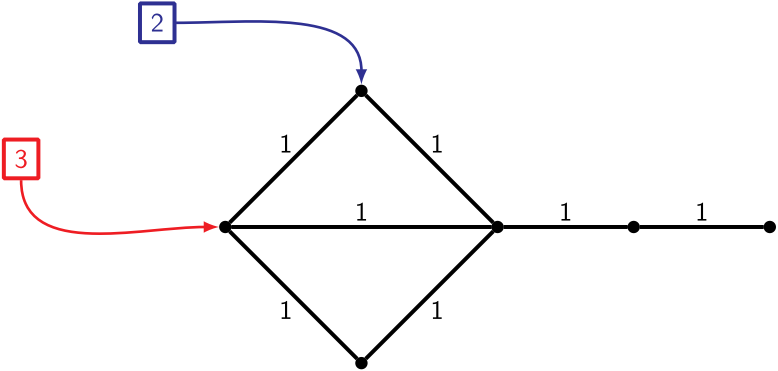
Sample example, 2nd try (again)



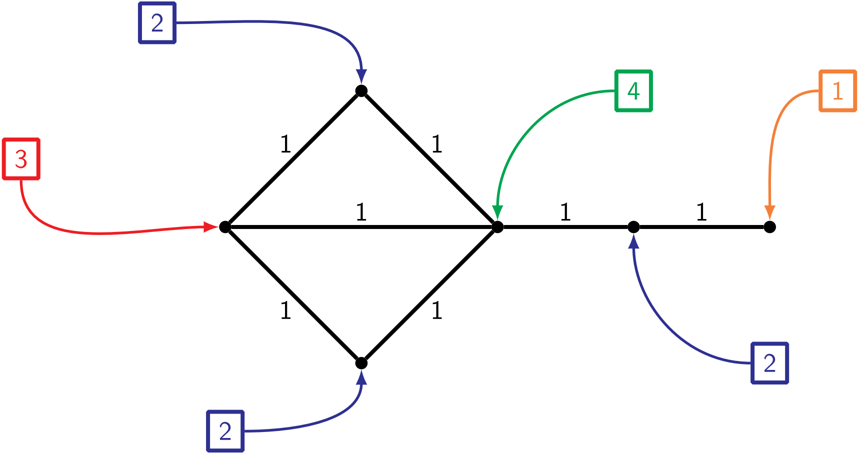
Sample example, 2nd try (again)



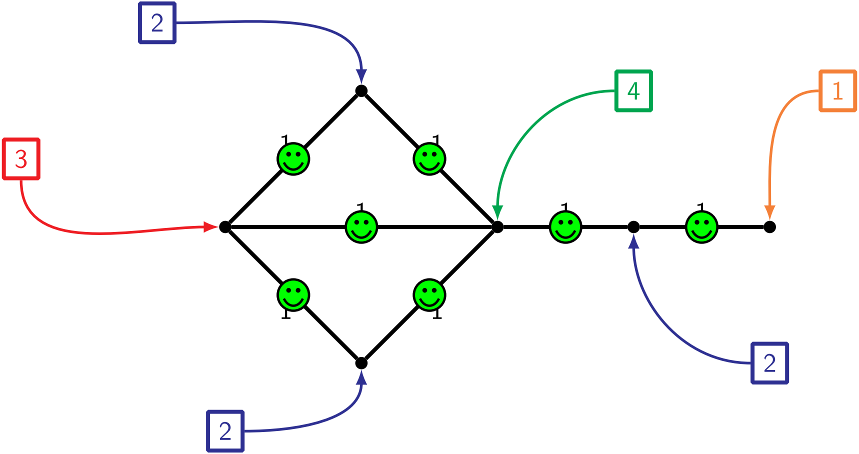
Sample example, 2nd try (again)



Sample example, 2nd try (again)



Sample example, 2nd try (again)



Early remarks

- K_2 is the **only** connected graph that does not admit such *proper labellings*

Early remarks

- K_2 is the **only** connected graph that does not admit such *proper labellings*
- For all other graphs, assign $1, \dots, k$ as desired, with k **as small as possible**?

Early remarks

- K_2 is the **only** connected graph that does not admit such *proper labellings*
- For all other graphs, assign $1, \ldots, k$ as desired, with k **as small as possible**?

1-2-3 Conjecture (Karoński, Łuczak, Thomason, 2004)

This is always possible with $k \leq 3$.

Most of what we know on the 1-2-3 Conjecture

- **Verification of the conjecture:**
  - mainly for complete graphs and 3-colourable graphs
  - other partial classes...

Most of what we know on the 1-2-3 Conjecture

- **Verification of the conjecture:**
  - mainly for complete graphs and 3-colourable graphs
  - other partial classes...
- **Complexity aspects:**
  - Deciding if $1,2$ suffice is NP-hard, but...
  - ... polytime solvable for bipartite graphs
  - bipartite graphs needing $1,2,3$ are the so-called *odd multi-cacti*

Most of what we know on the 1-2-3 Conjecture

- **Verification of the conjecture:**
  - mainly for complete graphs and 3-colourable graphs
  - other partial classes...
- **Complexity aspects:**
  - Deciding if 1,2 suffice is NP-hard, but...
  - ... polytime solvable for bipartite graphs
  - bipartite graphs needing 1,2,3 are the so-called *odd multi-cacti*
- **Approaching the conjecture:**
  - Best result to date: 1,2,3,4,5 suffice for all graphs
  - Better result: 1,2,3,4 suffice when regular or 4-chromatic

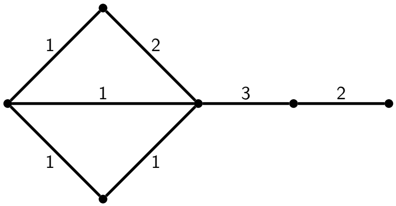
Most of what we know on the 1-2-3 Conjecture

- **Verification of the conjecture:**
  - mainly for complete graphs and 3-colourable graphs
  - other partial classes...
- **Complexity aspects:**
  - Deciding if $1, 2$ suffice is NP-hard, but...
  - ... polytime solvable for bipartite graphs
  - bipartite graphs needing $1, 2, 3$ are the so-called *odd multi-cacti*
- **Approaching the conjecture:**
  - Best result to date: $1, 2, 3, 4, 5$ suffice for all graphs
  - Better result: $1, 2, 3, 4$ suffice when regular or 4-chromatic

Also, many **side aspects, variants, etc.**

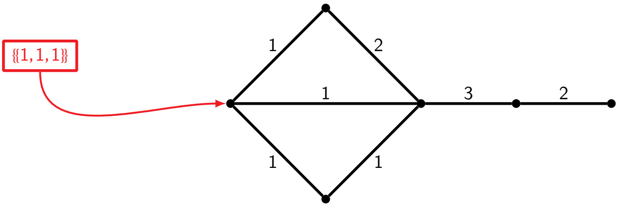
Speaking of variants...

- **Multiset variant**



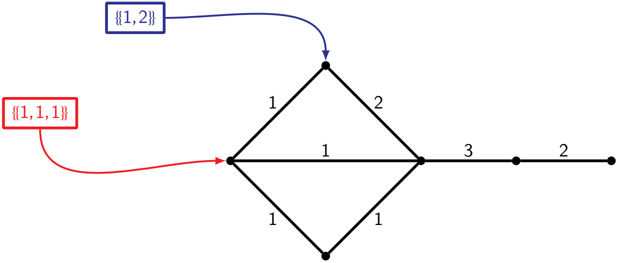
Speaking of variants...

- **Multiset variant**



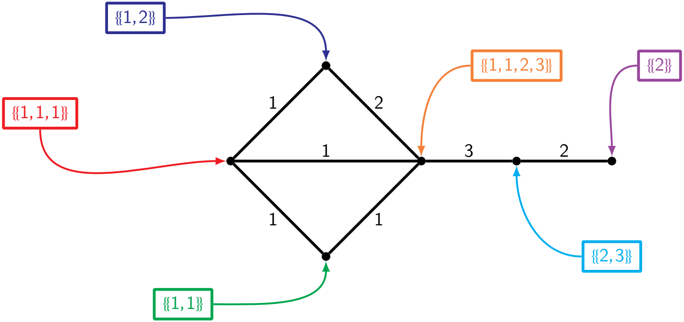
Speaking of variants...

- **Multiset variant**



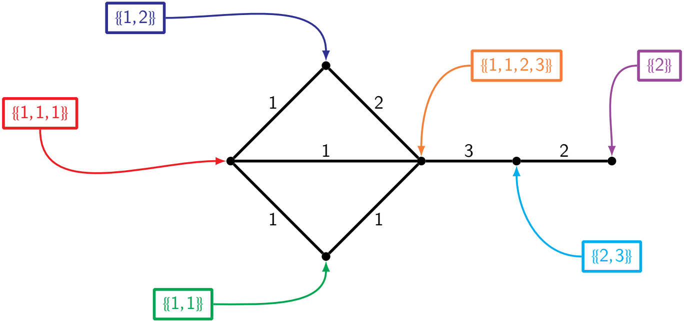
Speaking of variants...

- **Multiset variant**



Speaking of variants...

- **Multiset variant**

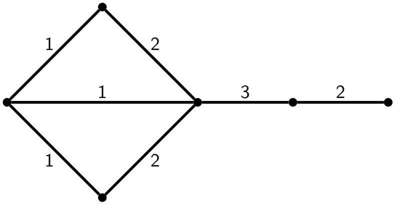


1-2-3 Conjecture, multiset version (Addario-Berry *et al.*, 2005)

Labels $1, 2, 3$ suffice for all graphs.

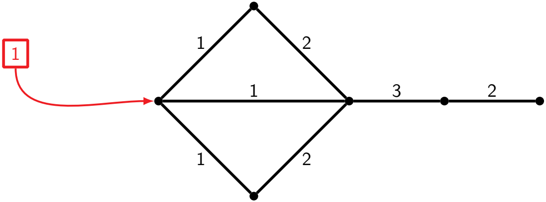
Speaking of variants...

- **Product variant**



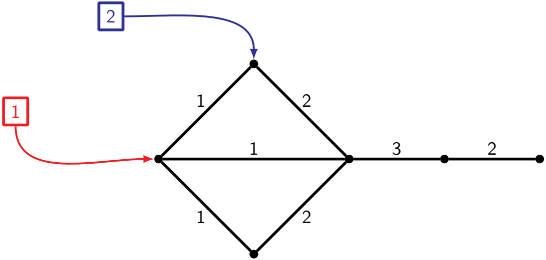
Speaking of variants...

- **Product variant**

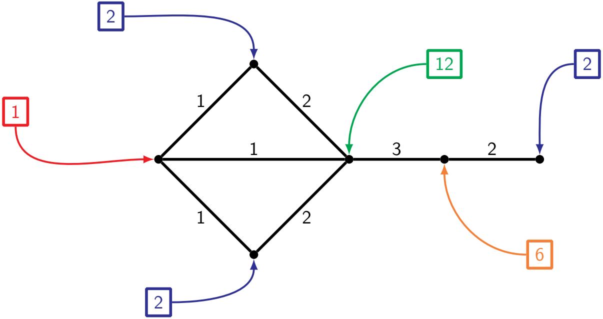


Speaking of variants...

- **Product variant**

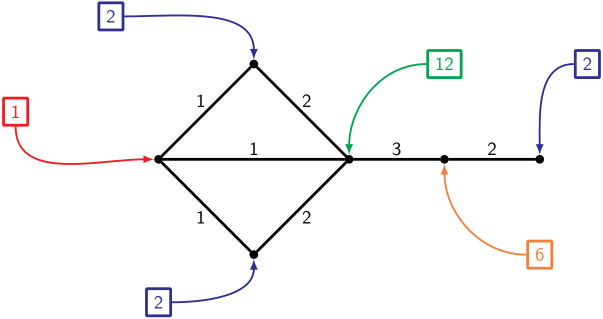


- **Product variant**



Speaking of variants...

- **Product variant**

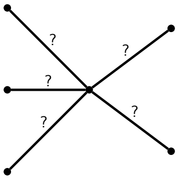


1-2-3 Conjecture, product version (Skowronek-Kaziów, 2012)

Labels 1, 2, 3 suffice for all graphs.

Comparing the three variants

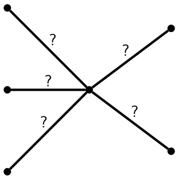
Labels are anything in $\{1, 2, 3\}$



If I tell you:

Comparing the three variants

Labels are anything in $\{1,2,3\}$

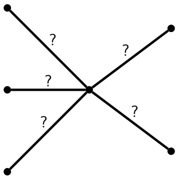


If I tell you:

- sum is 10

Comparing the three variants

Labels are anything in $\{1, 2, 3\}$

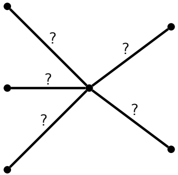


If I tell you:

- sum is 10 ☹ ☹ ☹
- product is 18 😐...

Comparing the three variants

Labels are anything in $\{1, 2, 3\}$

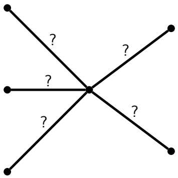


If I tell you:

- sum is 10 ☹ ☹ ☹
- product is 18 😐...oh wait... ☺

Comparing the three variants

Labels are anything in $\{1,2,3\}$

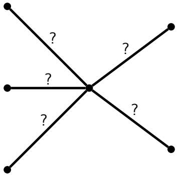


If I tell you:

- sum is 10 ☹ ☹ ☹
- product is 18 😐...oh wait... ☺ ... but meh 😐

Comparing the three variants

Labels are anything in $\{1, 2, 3\}$

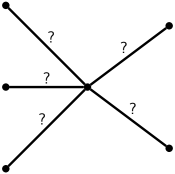


If I tell you:

- sum is 10 ☹ ☹ ☹
- product is 18 😐...oh wait... ☺ ... but meh 😐
- multiset is $\{\{1, 1, 2, 3, 3\}\}$ ☺ ☺ ☺ ☺ ☺

Comparing the three variants

Labels are anything in $\{1, 2, 3\}$



If I tell you:

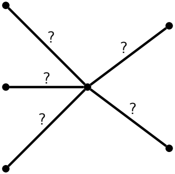
- sum is 10 🙁 🙁 🙁
- product is 18 😐...oh wait... 🙂 ... but meh 😐
- multiset is $\{\{1, 1, 2, 3, 3\}\}$ 🙂 🙂 🙂 🙂 🙂

Nice stuff:

- different sums or products $\Rightarrow$ different multisets

Comparing the three variants

Labels are anything in $\{1, 2, 3\}$



If I tell you:

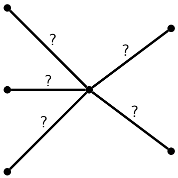
- sum is 10 ☹ ☹ ☹
- product is 18 😐...oh wait... ☺ ... but meh 😐
- multiset is $\{\{1, 1, 2, 3, 3\}\}$ ☺ ☺ ☺ ☺ ☺

Nice stuff:

- different sums or products ⇒ different multisets
- different degrees ⇒ different multisets

Comparing the three variants

Labels are anything in $\{1, 2, 3\}$



If I tell you:

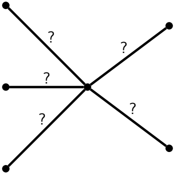
- sum is 10
- product is 18 ...oh wait... ... but meh
- multiset is $\{\{1, 1, 2, 3, 3\}\}$

Nice stuff:

- different sums or products $\Rightarrow$ different multisets
- different degrees $\Rightarrow$ different multisets
- in products, 2 and 3 are coprime, 1 is neutral:
  - 2 and 3 act similarly in products and multisets
  - 1 is like "skipping" labelling an edge

Comparing the three variants

Labels are anything in $\{1,2,3\}$



If I tell you:

- sum is 10 ☹ ☹ ☹
- product is 18 😐...oh wait... ☺ ... but meh 😐
- multiset is $\{\{1,1,2,3,3\}\}$ ☺ ☺ ☺ ☺ ☺

Nice stuff:

- different sums or products $\Rightarrow$ different multisets
- different degrees $\Rightarrow$ different multisets
- in products, 2 and 3 are coprime, 1 is neutral:
  - 2 and 3 act similarly in products and multisets
  - 1 is like "skipping" labelling an edge

$\Rightarrow$ **product version $\sim$ multiset version with a neutral label**

Progress towards the multiset and product versions

sum version $\gg$ **product version** $>$ **multiset version**

Progress towards the multiset and product versions

sum version $\gg$ product version > multiset version

- everything in the sum or product version applies in the multiset version

Progress towards the multiset and product versions

sum version $\gg$ product version $>$ multiset version

- everything in the sum or product version applies in the multiset version
- anything on the multiset version might give ideas for the product version

Progress towards the multiset and product versions

sum version $\gg$ product version $>$ multiset version

- everything in the sum or product version applies in the multiset version
- anything on the multiset version might give ideas for the product version

- Addario-Berry *et al.* (2005): $1, 2, 3, 4$ work for multisets

Progress towards the multiset and product versions

sum version $\gg$ product version $>$ multiset version

- everything in the sum or product version applies in the multiset version
- anything on the multiset version might give ideas for the product version

- Addario-Berry *et al.* (2005): $1, 2, 3, 4$ work for multisets
- Skowronek-Kaziów (2012): same for products, $1, 2, 3$ when $\chi \leq 3$ (as for sums)

Progress towards the multiset and product versions

sum version $\gg$ product version $>$ multiset version

- everything in the sum or product version applies in the multiset version
- anything on the multiset version might give ideas for the product version

- Addario-Berry *et al.* (2005): $1, 2, 3, 4$ work for multisets
- Skowronek-Kaziów (2012): same for products, $1, 2, 3$ when $\chi \leq 3$ (as for sums)
- ...

Progress towards the multiset and product versions

sum version $\gg$ product version $>$ multiset version

- everything in the sum or product version applies in the multiset version
- anything on the multiset version might give ideas for the product version

- Addario-Berry *et al.* (2005): $1, 2, 3, 4$ work for multisets
- Skowronek-Kaziów (2012): same for products, $1, 2, 3$ when $\chi \leq 3$ (as for sums)
- ...
- **Vučković (2018): multiset version is true!** ☺

Progress towards the multiset and product versions

sum version $\gg$ product version $>$ multiset version

- everything in the sum or product version applies in the multiset version
- anything on the multiset version might give ideas for the product version

- Addario-Berry *et al.* (2005): $1,2,3,4$ work for multisets
- Skowronek-Kaziów (2012): same for products, $1,2,3$ when $\chi \leq 3$ (as for sums)
- ...
- **Vučković (2018): multiset version is true!** ☺
- B., Hocquard, Lajou, Sopena (2021): product version when regular or $\chi = 4$

Progress towards the multiset and product versions

sum version $\gg$ product version $>$ multiset version

- everything in the sum or product version applies in the multiset version
- anything on the multiset version might give ideas for the product version

- Addario-Berry *et al.* (2005): $1,2,3,4$ work for multisets
- Skowronek-Kaziów (2012): same for products, $1,2,3$ when $\chi \leq 3$ (as for sums)
- ...
- **Vučković (2018): multiset version is true!** ☺
- B., Hocquard, Lajou, Sopena (2021): product version when regular or $\chi = 4$
- **B., Hocquard, Lajou, Sopena (2021+): product version is true!!** ☺

Progress towards the multiset and product versions

sum version $\gg$ product version $>$ multiset version

- everything in the sum or product version applies in the multiset version
- anything on the multiset version might give ideas for the product version

- Addario-Berry *et al.* (2005): 1, 2, 3, 4 work for multisets
- Skowronek-Kaziów (2012): same for products, 1, 2, 3 when $\chi \leq 3$ (as for sums)
- ...
- **Vučković (2018): multiset version is true!** ☺
- B., Hocquard, Lajou, Sopena (2021): product version when regular or $\chi = 4$
- **B., Hocquard, Lajou, Sopena (2021+): product version is true!!** ☺

For today: most of the proof!

Some terminology and conventions

For any 3-labelling ℓ:

- / = 1, / = 2, / = 3 (**Note:** / and / can be interchanged)

Some terminology and conventions

For any 3-labelling ℓ:

- / = 1, / = 2, / = 3 (**Note:** / and / can be interchanged)
- 2-degree $d_2(v)$ = number of 2's incident
- 3-degree $d_3(v)$ = number of 3's incident

Some terminology and conventions

For any 3-labelling ℓ:

- / = 1, / = 2, / = 3 (**Note:** / and / can be interchanged)

- 2-degree $d_2(v)$ = number of 2's incident
- 3-degree $d_3(v)$ = number of 3's incident

- G = 1-*monochromatic* (product is 1)
- R = 2-*monochromatic* (product is 2^p for $p > 0$)
- B = 3-*monochromatic* (product is 3^q for $q > 0$)

Some terminology and conventions

For any 3-labelling ℓ:

- / $= 1$, / $= 2$, / $= 3$ (**Note:** / and / can be interchanged)
- 2-degree $d_2(v)$ = number of 2's incident
- 3-degree $d_3(v)$ = number of 3's incident
- G = 1-*monochromatic* (product is 1)
- R = 2-*monochromatic* (product is 2^p for $p > 0$)
- B = 3-*monochromatic* (product is 3^q for $q > 0$)
- *bichromatic* = product is $2^p 3^q$ for $p, q > 0$

Some terminology and conventions

For any 3-labelling ℓ:

- / = 1, / = 2, / = 3 (**Note:** / and / can be interchanged)

- 2-degree $d_2(v)$ = number of 2's incident
- 3-degree $d_3(v)$ = number of 3's incident

- G = 1-*monochromatic* (product is 1)
- R = 2-*monochromatic* (product is 2^p for $p > 0$)
- B = 3-*monochromatic* (product is 3^q for $q > 0$)
- *bichromatic* = product is $2^p 3^q$ for $p, q > 0$

Remark: no conflict between

- i-monochromatic and j-monochromatic for $i \neq j$
- monochromatic and bichromatic

Actually, conflict between $2^p 3^q$ and $2^{p'} 3^{q'}$ iff $p = p'$ and $q = q'$

Some terminology and conventions

For any 3-labelling ℓ:

- / = 1, / = 2, / = 3 (**Note:** / and / can be interchanged)
- 2-degree $d_2(v)$ = number of 2's incident
- 3-degree $d_3(v)$ = number of 3's incident
- G = 1-*monochromatic* (product is 1)
- R = 2-*monochromatic* (product is 2^p for $p > 0$)
- B = 3-*monochromatic* (product is 3^q for $q > 0$)
- *bichromatic* = product is $2^p 3^q$ for $p, q > 0$

Remark: no conflict between

- i-monochromatic and j-monochromatic for $i \neq j$
- monochromatic and bichromatic

Actually, conflict between $2^p 3^q$ and $2^{p'} 3^{q'}$ iff $p = p'$ and $q = q'$

- S = *special* (product is $2^{2p} 3$ for $p > 0$)

Sketch of the proof

Main labelling steps

Start from all edges labelled 1

Main labelling steps

Start from all edges labelled 1

1. Partition $V(G)$ into $V_1 \cup \cdots \cup V_t$ so that:
   - the V_i's are independent
   - every $v \in V_i$ with $i > 1$ has a neighbour in V_j for every $j < i$

Main labelling steps

Start from all edges labelled 1

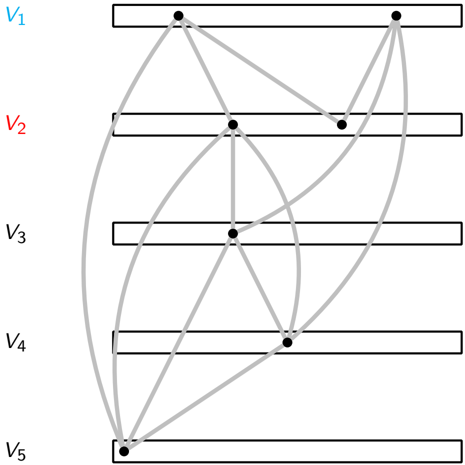
1. Partition $V(G)$ into $V_1 \cup \cdots \cup V_t$ so that:
   - the V_i's are independent
   - every $v \in V_i$ with $i > 1$ has a neighbour in V_j for every $j < i$
2. Relabel the upward edges of $V_3, \ldots, V_t$ to realise certain products

Main labelling steps

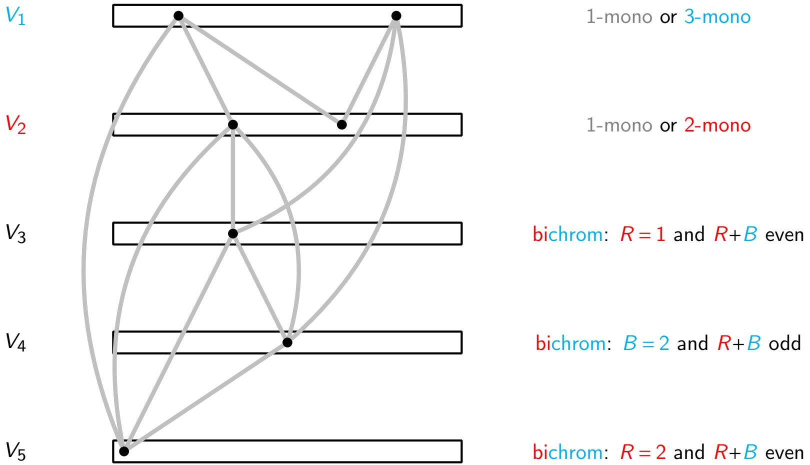
Start from all edges labelled 1

1. Partition $V(G)$ into $V_1 \cup \cdots \cup V_t$ so that:
   - the V_i's are independent
   - every $v \in V_i$ with $i > 1$ has a neighbour in V_j for every $j < i$
2. Relabel the upward edges of $V_3, \ldots, V_t$ to realise certain products
3. Get rid of conflicts in (V_1, V_2)

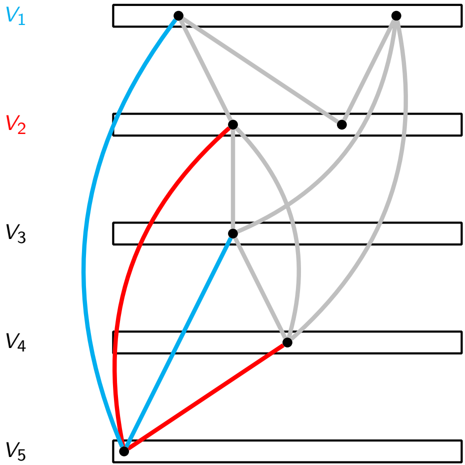
The type of labelling we want by the end of Step 2



The type of labelling we want by the end of Step 2



The type of labelling we want by the end of Step 2



V_1: 1-mono or 3-mono

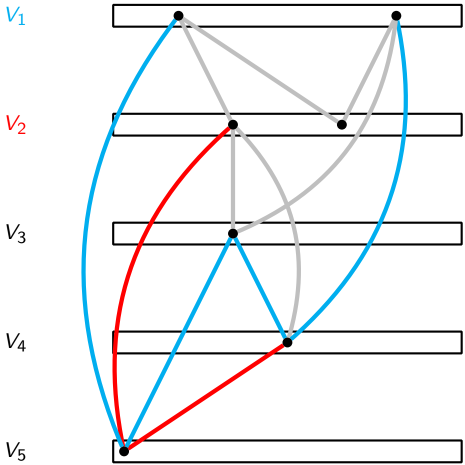
V_2: 1-mono or 2-mono

V_3: bichrom: $R = 1$ and $R+B$ even

V_4: bichrom: $B = 2$ and $R+B$ odd

V_5: bichrom: $R = 2$ and $R+B$ even

The type of labelling we want by the end of Step 2



1-mono or 3-mono

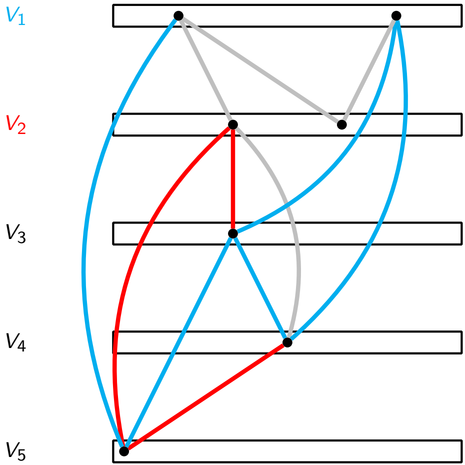
1-mono or 2-mono

bichrom: $R = 1$ and $R + B$ even

bichrom: $B = 2$ and $R + B$ odd

bichrom: $R = 2$ and $R + B$ even

The type of labelling we want by the end of Step 2



1-mono or 3-mono

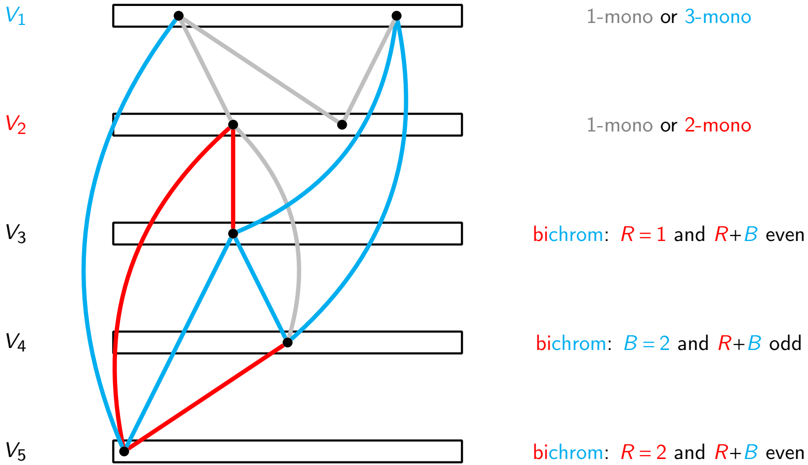
1-mono or 2-mono

bichrom: $R=1$ and $R+B$ even

bichrom: $B=2$ and $R+B$ odd

bichrom: $R=2$ and $R+B$ even

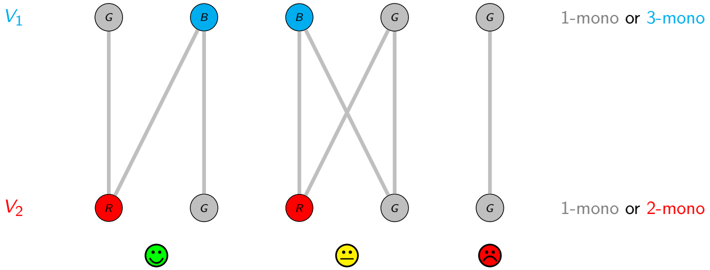
The type of labelling we want by the end of Step 2



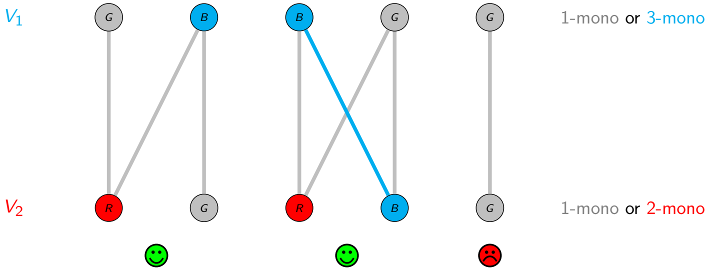
Note:

- no conflict between odd layers; same for even layers
- same between odd layers and even layers (except for 1-mono across (V_1, V_2))
- no special vertex ($B = 1$ and $R + B$ odd)

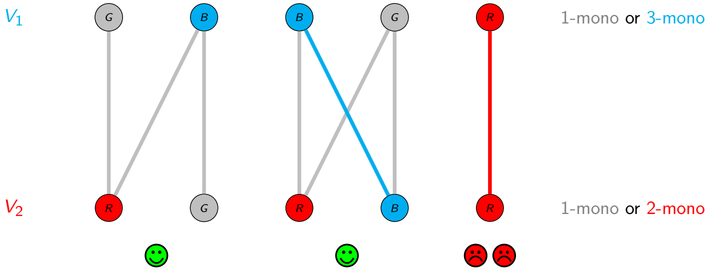
Getting rid of remaining conflicts in Step 3



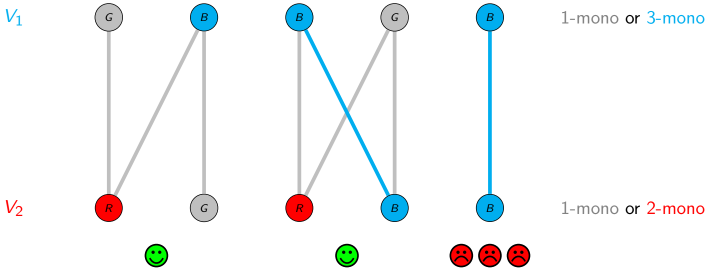
Getting rid of remaining conflicts in Step 3



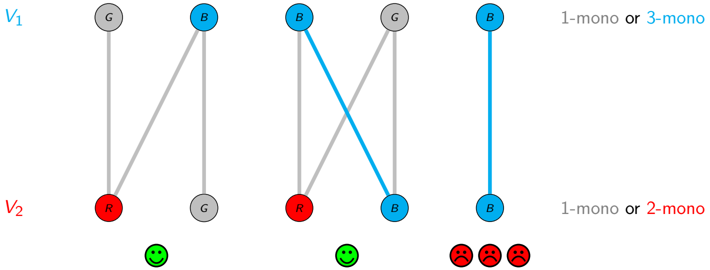
Getting rid of remaining conflicts in Step 3



Getting rid of remaining conflicts in Step 3



Getting rid of remaining conflicts in Step 3



Do not forget about $V_3, \ldots, V_t$!!

⇒ Keep vertices 1-mono, 2-mono, 3-mono, special

Main labelling steps – REVISITED

Start from all edges labelled 1

Main labelling steps – REVISITED

Start from all edges labelled 1

1. Partition $V(G)$ into $V_1 \cup \cdots \cup V_t$ so that:
   - the V_i's are independent
   - every $v \in V_i$ with $i > 1$ has a neighbour in V_j for every $j < i$

Main labelling steps – REVISITED

Start from all edges labelled 1

1. Partition $V(G)$ into $V_1 \cup \cdots \cup V_t$ so that:
   - the V_i's are independent
   - every $v \in V_i$ with $i > 1$ has a neighbour in V_j for every $j < i$
2. Relabel the upward edges of $V_3, \ldots, V_t$ so that
   - certain products are realised

Main labelling steps – REVISITED

Start from all edges labelled 1

1. Partition $V(G)$ into $V_1 \cup \cdots \cup V_t$ so that:
   - the V_i's are independent
   - every $v \in V_i$ with $i > 1$ has a neighbour in V_j for every $j < i$
2. Relabel the upward edges of $V_3, \ldots, V_t$ so that
   - certain products are realised
   - **no isolated 1-mono edge in** (V_1, V_2)

Main labelling steps – REVISITED

Start from all edges labelled 1

1. Partition $V(G)$ into $V_1 \cup \cdots \cup V_t$ so that:
   - the V_i's are independent
   - every $v \in V_i$ with $i > 1$ has a neighbour in V_j for every $j < i$
2. Relabel the upward edges of $V_3, \dots, V_t$ so that
   - certain products are realised
   - **no isolated 1-mono edge in (V_1, V_2)**
3. Get rid of conflicts in (V_1, V_2), playing with 1-mono, 2-mono, 3-mono, special

– Step 1 –

Getting a “good” partition $V_1 \cup \cdots \cup V_t$ of $V(G)$

Getting the upward edges property

- Pick V_1 independent **as big as possible**

Getting the upward edges property

- Pick V_1 independent **as big as possible**
- In $V(G) \setminus V_1$, pick V_2 independent **as big as possible**

Getting the upward edges property

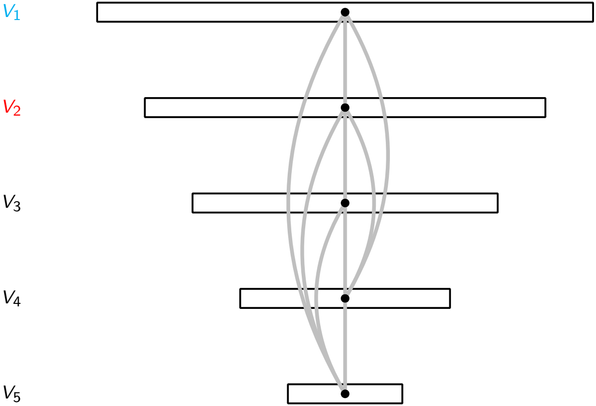
- Pick V_1 independent **as big as possible**
- In $V(G) \setminus V_1$, pick V_2 independent **as big as possible**
- In $V(G) \setminus (V_1 \cup V_2)$, pick V_3 independent **as big as possible**

Getting the upward edges property

- Pick V_1 independent **as big as possible**
- In $V(G) \setminus V_1$, pick V_2 independent **as big as possible**
- In $V(G) \setminus (V_1 \cup V_2)$, pick V_3 independent **as big as possible**
- Etc.

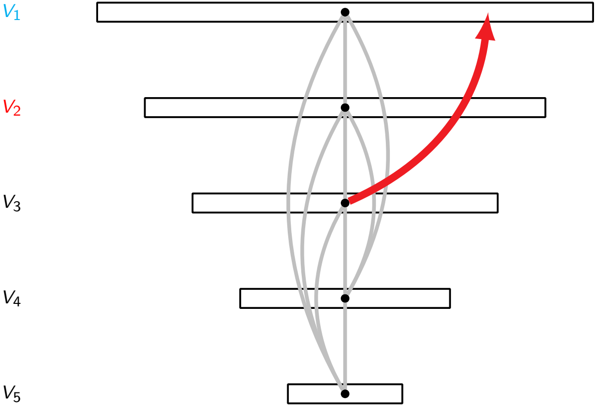
Getting the upward edges property

- Pick V_1 independent **as big as possible**
- In $V(G) \setminus V_1$, pick V_2 independent **as big as possible**
- In $V(G) \setminus (V_1 \cup V_2)$, pick V_3 independent **as big as possible**
- Etc.



Getting the upward edges property

- Pick V_1 independent **as big as possible**
- In $V(G) \setminus V_1$, pick V_2 independent **as big as possible**
- In $V(G) \setminus (V_1 \cup V_2)$, pick V_3 independent **as big as possible**
- Etc.



An additional swapping property

Lemma

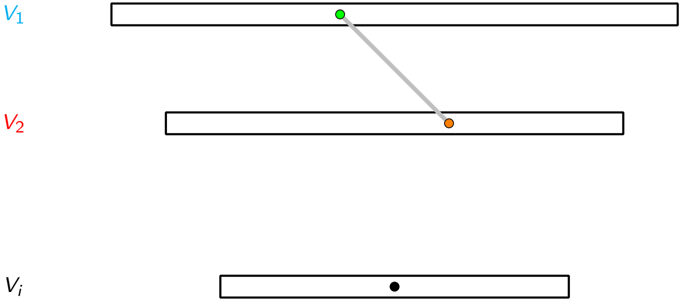
We can choose $V_1 \cup \cdots \cup V_t$ so that if $e = (u, v) \in (V_1, V_2)$ is isolated, then u and v can be freely exchanged between V_1 and V_2 without spoiling any of the desired properties (independence, upward edges, etc.).

An additional swapping property

Lemma

We can choose $V_1 \cup \cdots \cup V_t$ so that if $e = (u, v) \in (V_1, V_2)$ is isolated, then u and v can be freely exchanged between V_1 and V_2 without spoiling any of the desired properties (independence, upward edges, etc.).

Just choose V_1 and V_2 so that $V_1 \cup V_2$ as large as possible

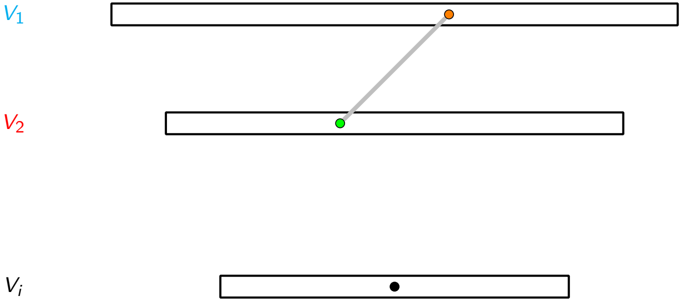


An additional swapping property

Lemma

We can choose $V_1 \cup \cdots \cup V_t$ so that if $e = (u, v) \in (V_1, V_2)$ is isolated, then u and v can be freely exchanged between V_1 and V_2 without spoiling any of the desired properties (independence, upward edges, etc.).

Just choose V_1 and V_2 so that $V_1 \cup V_2$ as large as possible

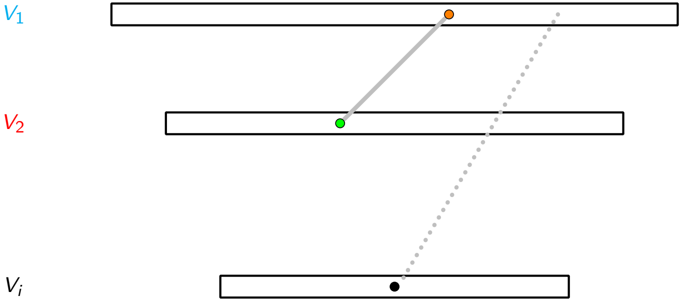


An additional swapping property

Lemma

We can choose $V_1 \cup \cdots \cup V_t$ so that if $e = (u, v) \in (V_1, V_2)$ is isolated, then u and v can be freely exchanged between V_1 and V_2 without spoiling any of the desired properties (independence, upward edges, etc.).

Just choose V_1 and V_2 so that $V_1 \cup V_2$ as large as possible

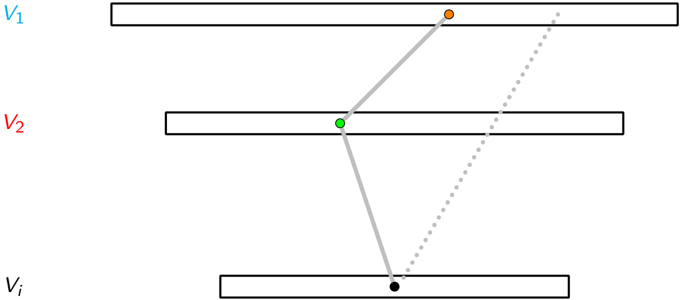


An additional swapping property

Lemma

We can choose $V_1 \cup \cdots \cup V_t$ so that if $e = (u,v) \in (V_1, V_2)$ is isolated, then u and v can be freely exchanged between V_1 and V_2 without spoiling any of the desired properties (independence, upward edges, etc.).

Just choose V_1 and V_2 so that $V_1 \cup V_2$ as large as possible

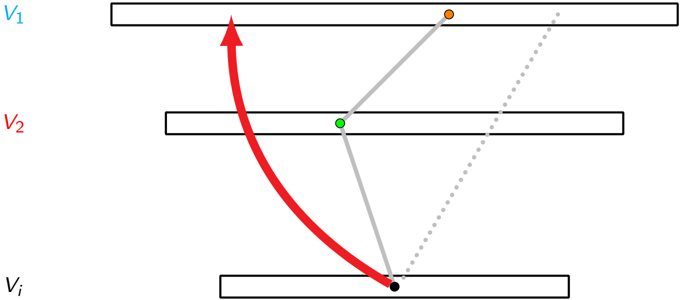


An additional swapping property

Lemma

We can choose $V_1 \cup \cdots \cup V_t$ so that if $e = (u, v) \in (V_1, V_2)$ is isolated, then u and v can be freely exchanged between V_1 and V_2 without spoiling any of the desired properties (independence, upward edges, etc.).

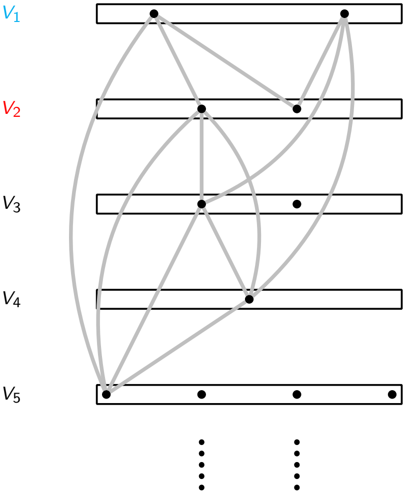
Just choose V_1 and V_2 so that $V_1 \cup V_2$ as large as possible



– Step 2 –

Relabelling the upward edges of $V_3, \ldots, V_t$

Recap of what is desired



1-mono or 3-mono

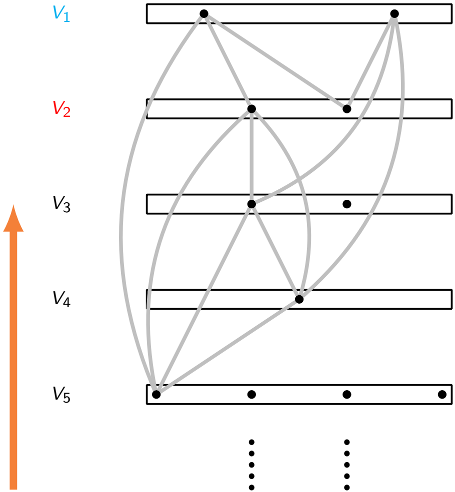
1-mono or 2-mono

bichrom: $R = 1$ and $R+B$ even

bichrom: $B = 2$ and $R+B$ odd

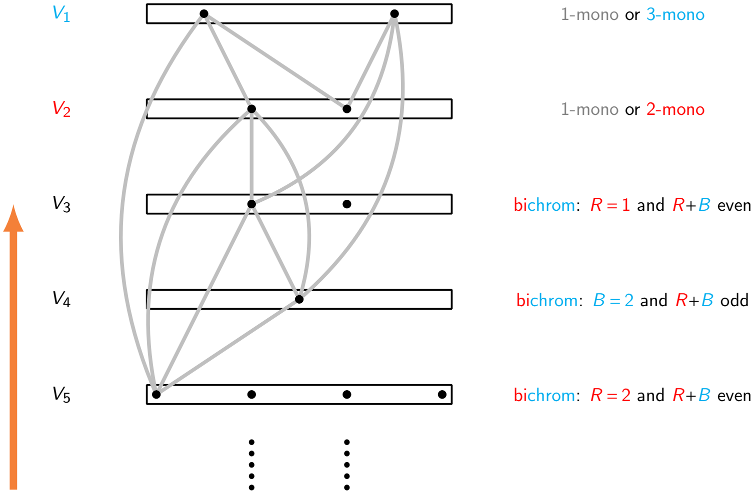
bichrom: $R = 2$ and $R+B$ even

Recap of what is desired



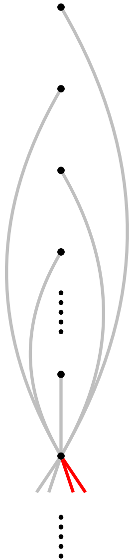
- V_1: 1-mono or 3-mono
- V_2: 1-mono or 2-mono
- V_3: bichrom: $R = 1$ and $R+B$ even
- V_4: bichrom: $B = 2$ and $R+B$ odd
- V_5: bichrom: $R = 2$ and $R+B$ even

Recap of what is desired



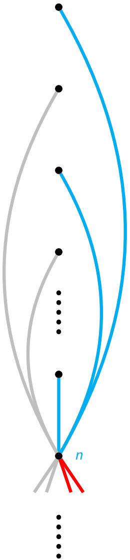
Watch out: even (odd, resp.) layers require a bounded number of 3's (2's, resp.)
⇒ even (odd, resp.) layers produce their 3's (2's, resp.) upwards
⇒ assume even (odd, resp.) layers do not receive 3's (2's, resp.) from below

Case of a vertex in some V_{2n}



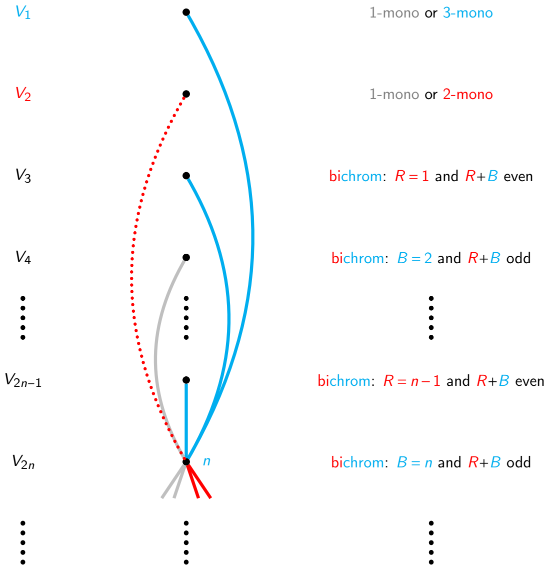
V_1	1-mono or 3-mono
V_2	1-mono or 2-mono
V_3	bichrom: $R = 1$ and $R + B$ even
V_4	bichrom: $B = 2$ and $R + B$ odd
$\vdots$	$\vdots$
V_{2n-1}	bichrom: $R = n - 1$ and $R + B$ even
V_{2n}	bichrom: $B = n$ and $R + B$ odd
$\vdots$	$\vdots$

Case of a vertex in some V_{2n}



V_1	1-mono or 3-mono
V_2	1-mono or 2-mono
V_3	bichrom: $R = 1$ and $R+B$ even
V_4	bichrom: $B = 2$ and $R+B$ odd
$\vdots$	$\vdots$
V_{2n-1}	bichrom: $R = n-1$ and $R+B$ even
V_{2n}	bichrom: $B = n$ and $R+B$ odd
$\vdots$	$\vdots$

Case of a vertex in some V_{2n} – fixing parity



Works because...

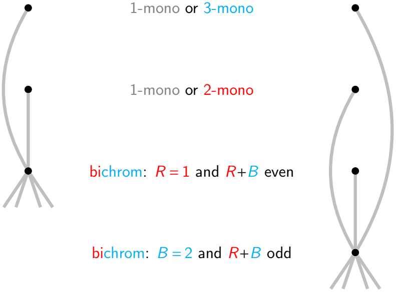
- Always have exactly the desired number of layers with distinct parity above
  $\Rightarrow$ get the required fixed number of labels (3 for even layers, 2 for odd layers)

Works because...

- Always have exactly the desired number of layers with distinct parity above
  ⇒ get the required fixed number of labels (3 for even layers, 2 for odd layers)
- Deep enough layers always have at least two layers with the same parity above
  ⇒ make sure vertices are bichrom and/or adjust parity of $R+B$

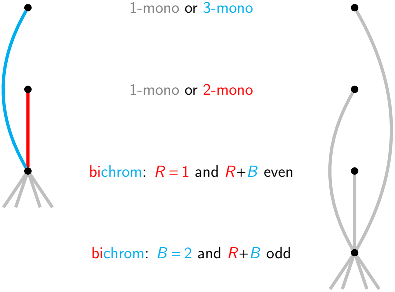
Works because...

- Always have exactly the desired number of layers with distinct parity above
  ⇒ get the required fixed number of labels (3 for even layers, 2 for odd layers)
- Deep enough layers always have at least two layers with the same parity above
  ⇒ make sure vertices are bichrom and/or adjust parity of $R+B$
- ... only V_3 and V_4 might be problematic...
  ... but actually things are (luckily!) fine!



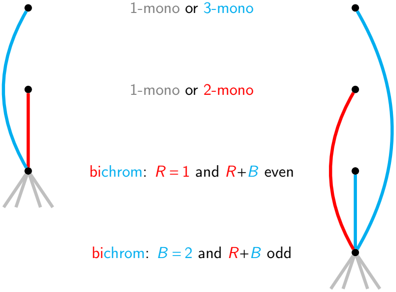
Works because...

- Always have exactly the desired number of layers with distinct parity above
  ⇒ get the required fixed number of labels (3 for even layers, 2 for odd layers)
- Deep enough layers always have at least two layers with the same parity above
  ⇒ make sure vertices are bichrom and/or adjust parity of $R+B$
- ... only V_3 and V_4 might be problematic...
  ... but actually things are (luckily!) fine!

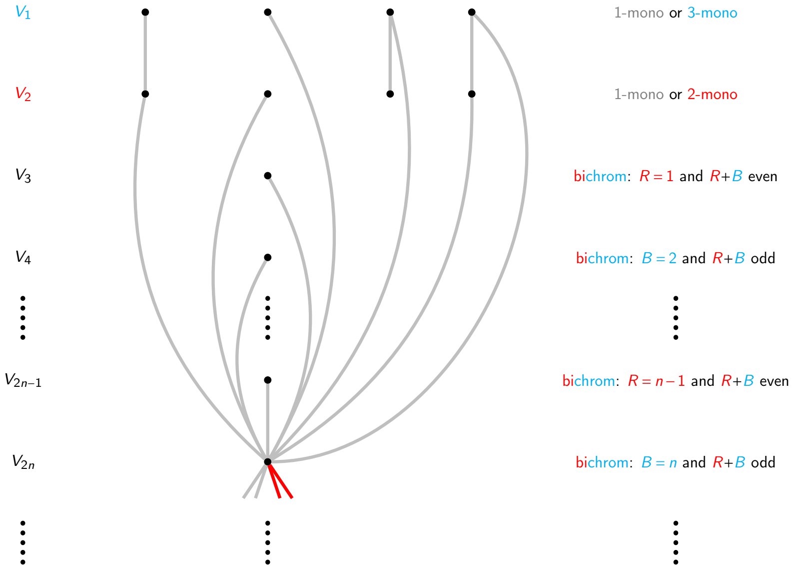


Works because...

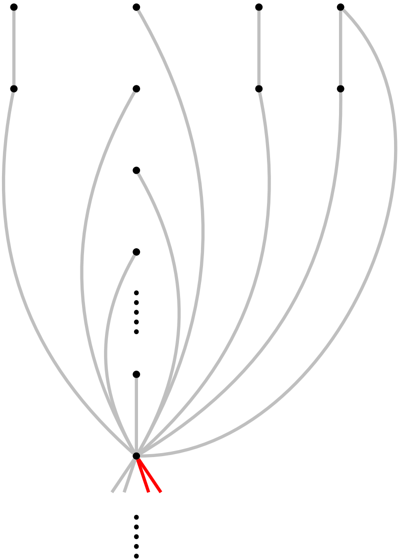
- Always have exactly the desired number of layers with distinct parity above
  ⇒ get the required fixed number of labels (3 for even layers, 2 for odd layers)
- Deep enough layers always have at least two layers with the same parity above
  ⇒ make sure vertices are bichrom and/or adjust parity of $R+B$
- ... only V_3 and V_4 might be problematic...
  ... but actually things are (luckily!) fine!



Watch out for adjacent isolated edges in (V_1, V_2)!

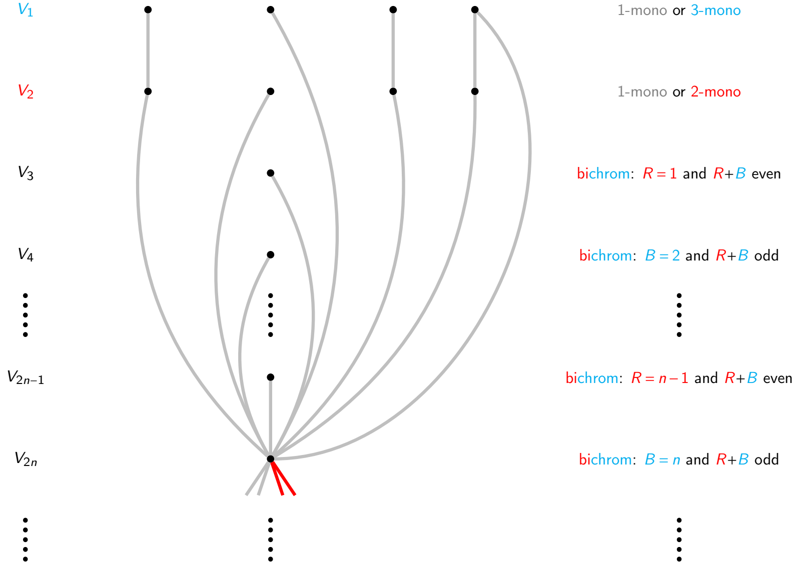


Swapping adjacent isolated edges

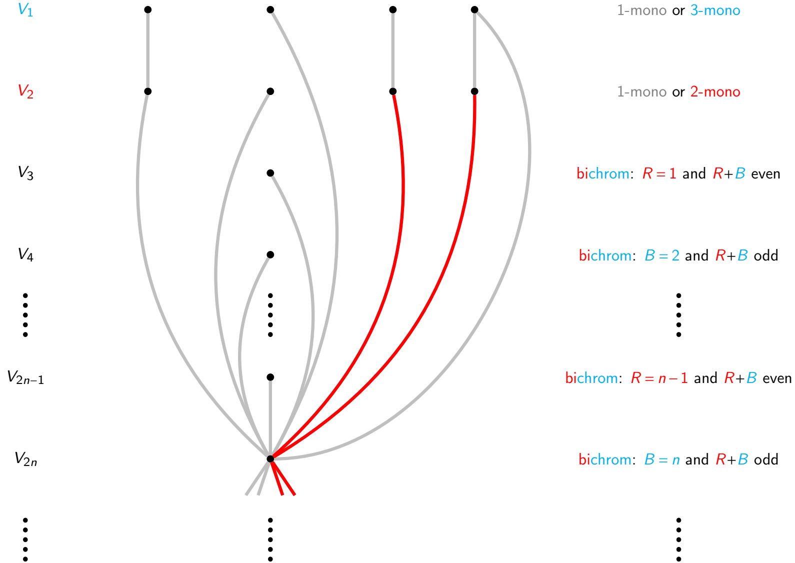


V_1	1-mono or 3-mono
V_2	1-mono or 2-mono
V_3	bichrom: $R=1$ and $R+B$ even
V_4	bichrom: $B=2$ and $R+B$ odd
⋮	⋮
V_{2n-1}	bichrom: $R=n-1$ and $R+B$ even
V_{2n}	bichrom: $B=n$ and $R+B$ odd
⋮	⋮

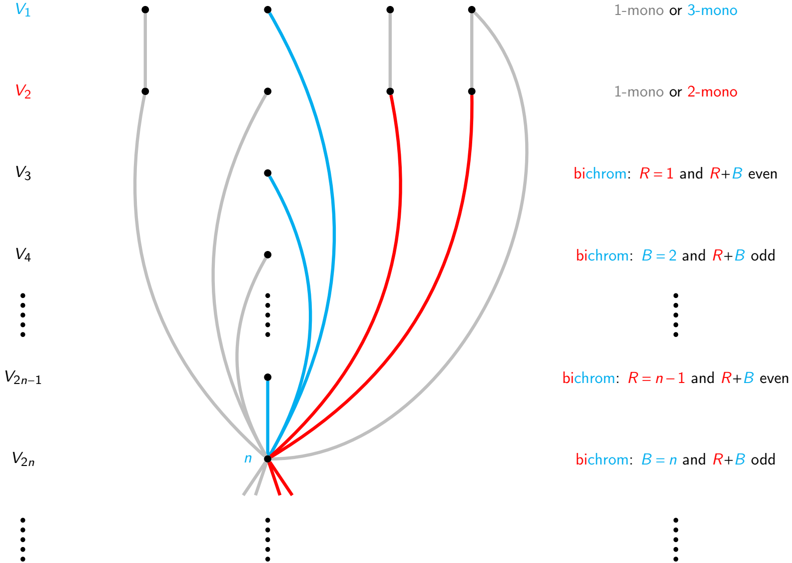
Making adjacent isolated edges happy



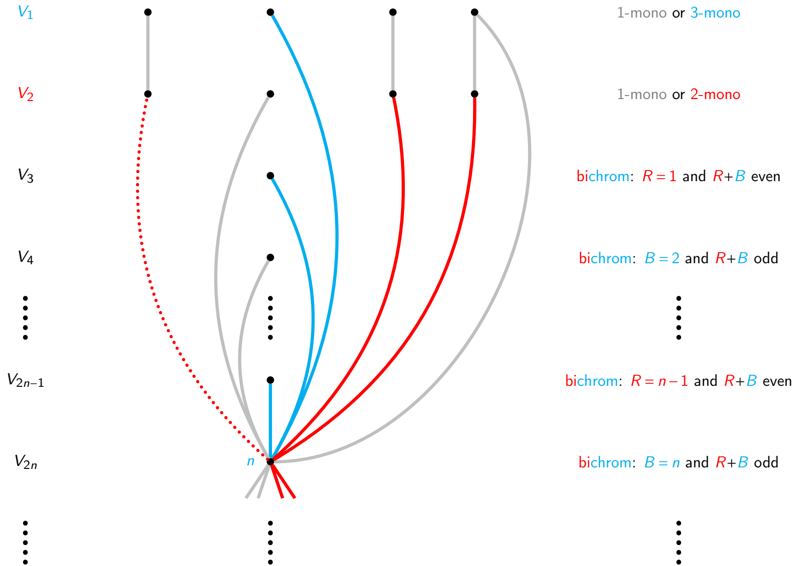
Making adjacent isolated edges happy



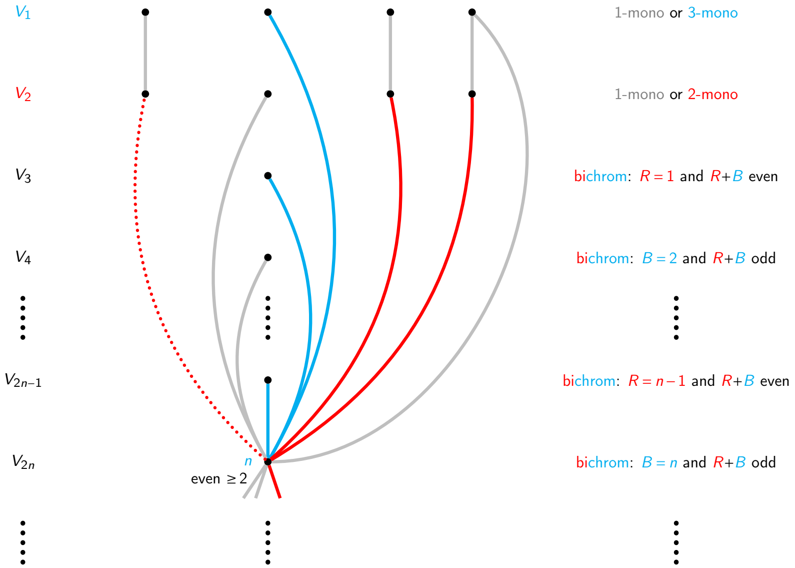
Making adjacent isolated edges happy



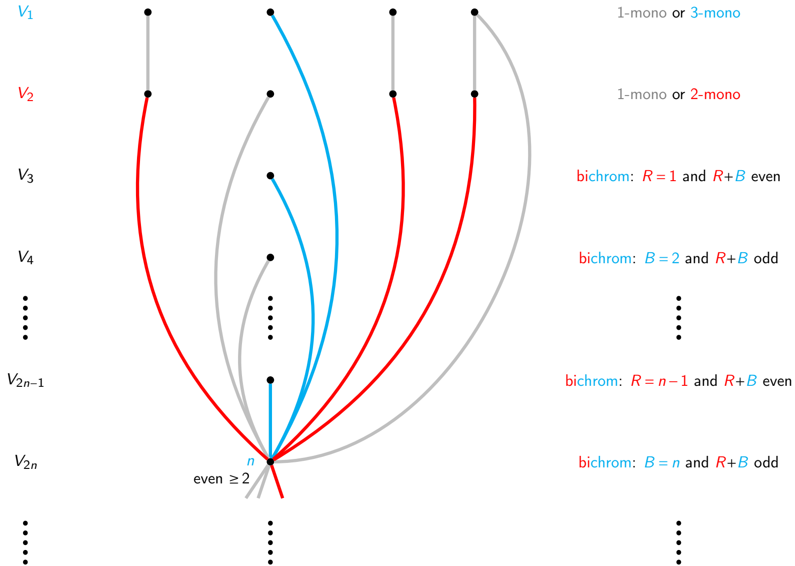
Making adjacent isolated edges happy – fixing parity



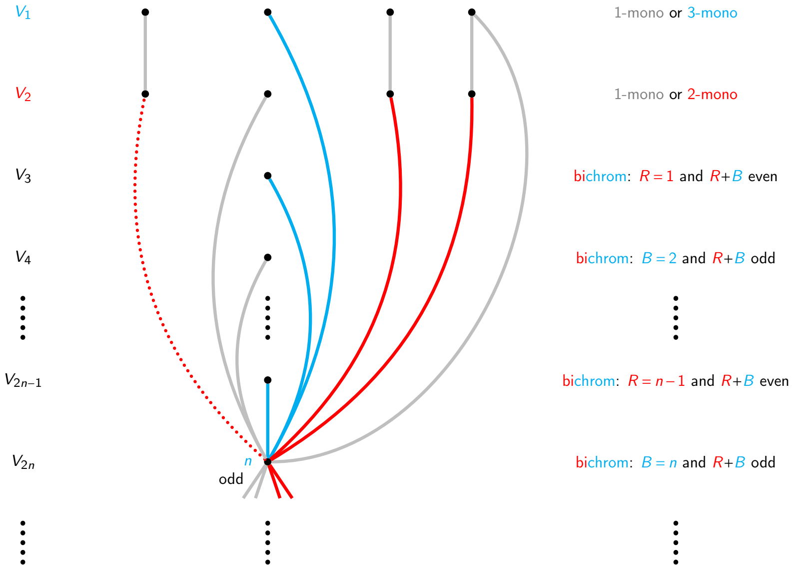
Making adjacent isolated edges happy – fixing parity



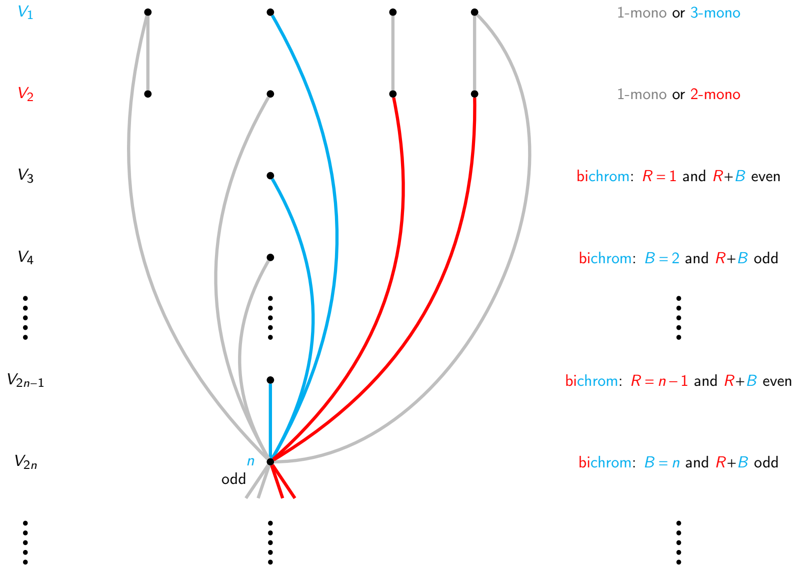
Making adjacent isolated edges happy – fixing parity



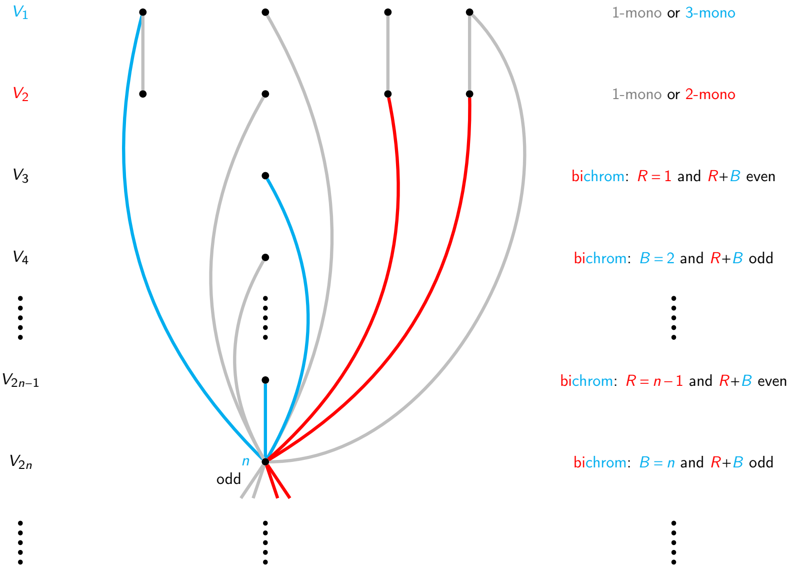
Making adjacent isolated edges happy – fixing parity



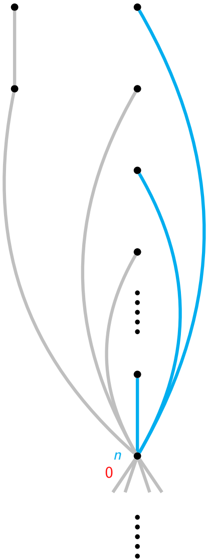
Making adjacent isolated edges happy – fixing parity



Making adjacent isolated edges happy – fixing parity

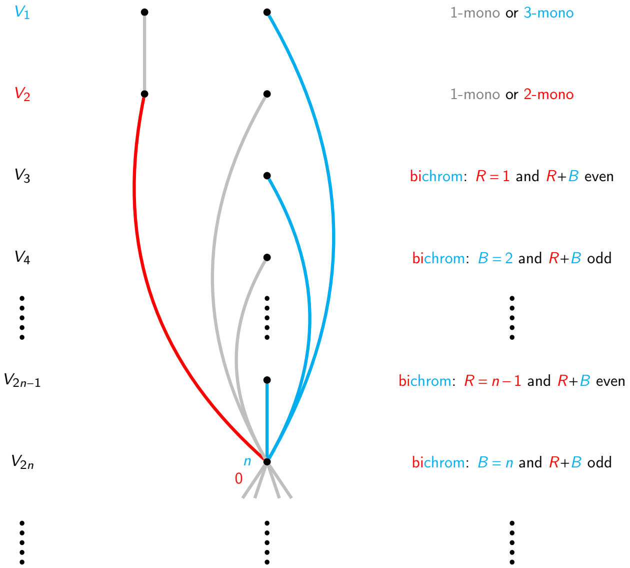


Making adjacent isolated edges happy – fixing parity

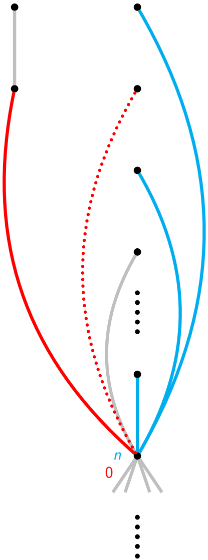


V_1	1-mono or 3-mono
V_2	1-mono or 2-mono
V_3	bichrom: $R=1$ and $R+B$ even
V_4	bichrom: $B=2$ and $R+B$ odd
⋮	⋮
V_{2n-1}	bichrom: $R=n-1$ and $R+B$ even
V_{2n}	bichrom: $B=n$ and $R+B$ odd
⋮	⋮

Making adjacent isolated edges happy – fixing parity



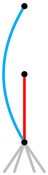
Making adjacent isolated edges happy – fixing parity



V_1	1-mono or 3-mono
V_2	1-mono or 2-mono
V_3	bichrom: $R = 1$ and $R+B$ even
V_4	bichrom: $B = 2$ and $R+B$ odd
⋮	⋮
V_{2n-1}	bichrom: $R = n-1$ and $R+B$ even
V_{2n}	bichrom: $B = n$ and $R+B$ odd
⋮	⋮

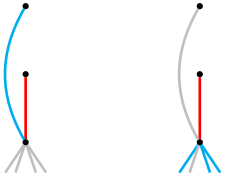
More examples – The (slightly more) intricate case of V_3

Want: bichrom: $R=1$ and $R+B$ even



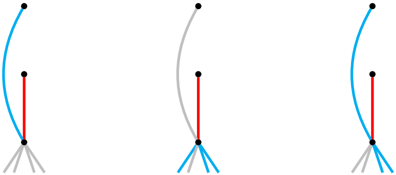
More examples – The (slightly more) intricate case of V_3

Want: bichrom: $R=1$ and $R+B$ even



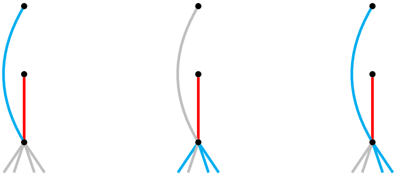
More examples – The (slightly more) intricate case of V_3

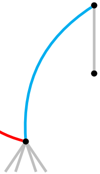
Want: bichrom: $R = 1$ and $R+B$ even



More examples – The (slightly more) intricate case of V_3

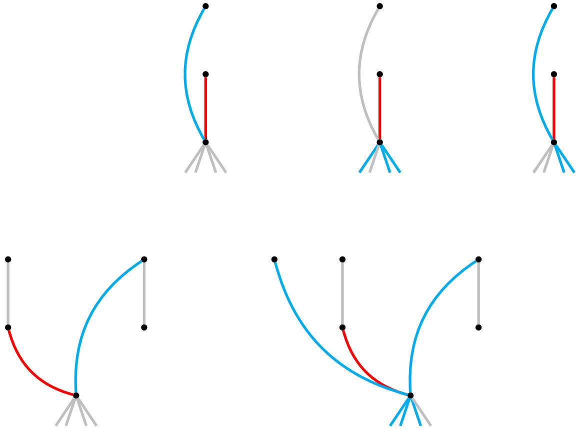
Want: bichrom: $R = 1$ and $R + B$ even





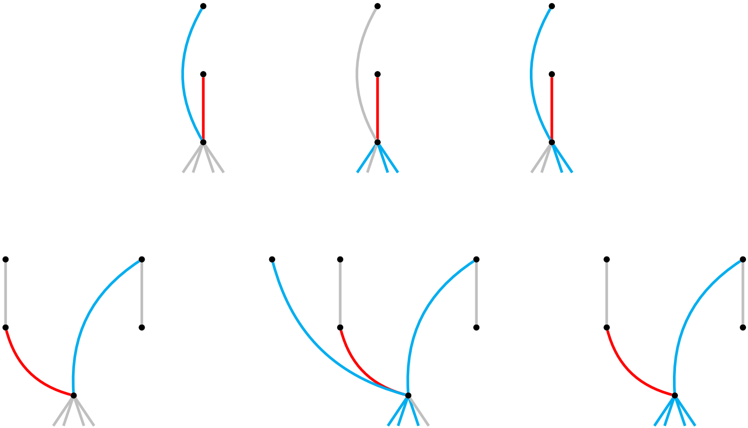
More examples – The (slightly more) intricate case of V_3

Want: bichrom: $R = 1$ and $R + B$ even



More examples – The (slightly more) intricate case of V_3

Want: bichrom: $R = 1$ and $R+B$ even

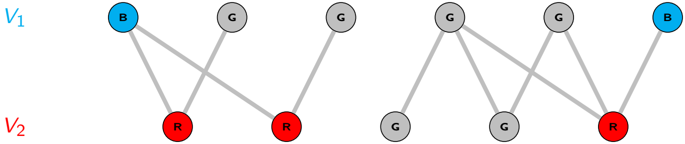


– Step 3 –

Getting rid of conflicts in (V_1, V_2)

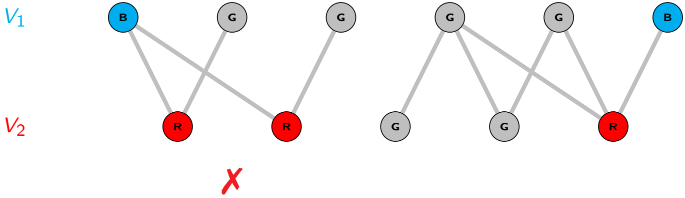
Getting rid of conflicts one by one

$\mathscr{H}$: components of $G[V_1 \cup V_2]$ having conflicting (1-mono) vertices



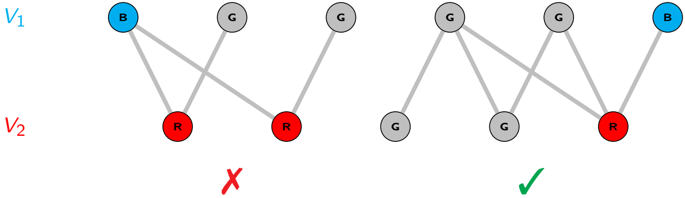
Getting rid of conflicts one by one

$\mathscr{H}$: components of $G[V_1 \cup V_2]$ having conflicting (1-mono) vertices



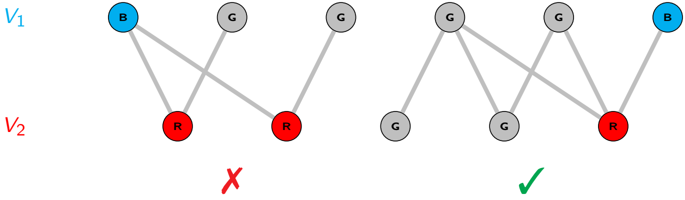
Getting rid of conflicts one by one

$\mathscr{H}$: components of $G[V_1 \cup V_2]$ having conflicting (1-mono) vertices



Getting rid of conflicts one by one

$\mathscr{H}$: components of $G[V_1 \cup V_2]$ having conflicting (1-mono) vertices

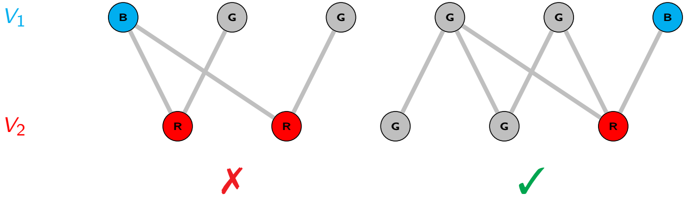


Deal with every $H \in \mathscr{H}$:

- 1-mono, 2-mono, 3-mono, special ⇒ no conflicts with $V_3, \ldots, V_t$
- **Remark:** H's can be treated independently

Getting rid of conflicts one by one

$\mathscr{H}$: components of $G[V_1 \cup V_2]$ having conflicting (1-mono) vertices



Deal with every $H \in \mathscr{H}$:

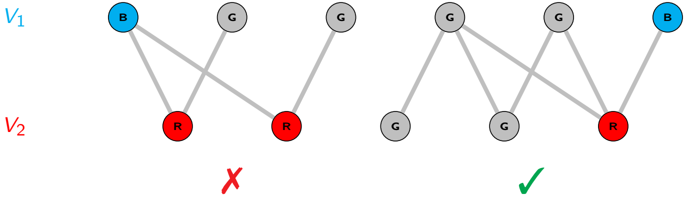
- 1-mono, 2-mono, 3-mono, special ⇒ no conflicts with $V_3, \ldots, V_t$
- **Remark:** H's can be treated independently

Several cases:

1. H contains either:
    - a 1-mono $v_1 \in V_1$ with two 1-mono degre-1 neighbours $u_1, u_2 \in V_2$, or
    - a 3-mono $v \in V_1$

Getting rid of conflicts one by one

$\mathcal{H}$: components of $G[V_1 \cup V_2]$ having conflicting (1-mono) vertices



Deal with every $H \in \mathcal{H}$:

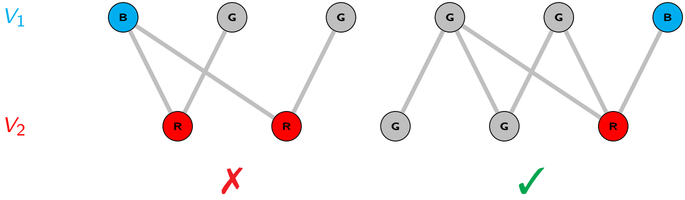
- 1-mono, 2-mono, 3-mono, special $\Rightarrow$ no conflicts with $V_3, \ldots, V_t$
- **Remark:** H's can be treated independently

Several cases:

1. H contains either:
   - a 1-mono $v_1 \in V_1$ with two 1-mono degre-1 neighbours $u_1, u_2 \in V_2$, or
   - a 3-mono $v \in V_1$
2. H contains a 1-mono $u \in V_2$ with at least two neighbours in H

Getting rid of conflicts one by one

$\mathcal{H}$: components of $G[V_1 \cup V_2]$ having conflicting (1-mono) vertices



Deal with every $H \in \mathcal{H}$:

- 1-mono, 2-mono, 3-mono, special $\Rightarrow$ no conflicts with $V_3, \ldots, V_t$
- **Remark:** H's can be treated independently

Several cases:

1. H contains either:
   - a 1-mono $v_1 \in V_1$ with two 1-mono degre-1 neighbours $u_1, u_2 \in V_2$, or
   - a 3-mono $v \in V_1$
2. H contains a 1-mono $u \in V_2$ with at least two neighbours in H
3. H contains none of the previous

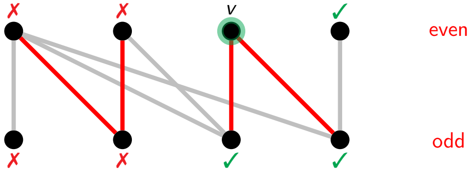
A useful lemma

Lemma

Let $s \in \{2,3\}$, and let H be a connected bipartite graph whose edges are labelled 1 or s. Consider any vertex v in any part $V_i \in \{V_1, V_2\}$ of H. We can relabel the edges of H with 1 and s so that:

- $d_s(u)$ is odd (even, resp.) for every $u \in V_i \setminus \{v\}$, and
- $d_s(u)$ is even (odd, resp.) for every $u \in V_{3-i}$.

Note: Remains true, even if some "contribution" from outside



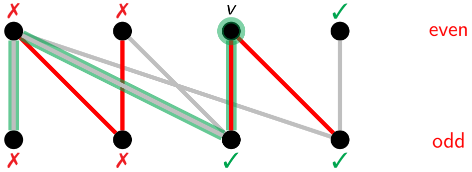
A useful lemma

Lemma

Let $s \in \{2,3\}$, and let H be a connected bipartite graph whose edges are labelled 1 or s. Consider any vertex v in any part $V_i \in \{V_1, V_2\}$ of H. We can relabel the edges of H with 1 and s so that:

- $d_s(u)$ is odd (even, resp.) for every $u \in V_i \setminus \{v\}$, and
- $d_s(u)$ is even (odd, resp.) for every $u \in V_{3-i}$.

Note: Remains true, even if some "contribution" from outside



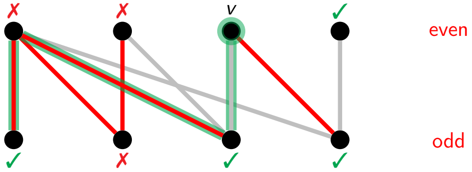
A useful lemma

Lemma

Let $s \in \{2,3\}$, and let H be a connected bipartite graph whose edges are labelled 1 or s. Consider any vertex v in any part $V_i \in \{V_1, V_2\}$ of H. We can relabel the edges of H with 1 and s so that:

- $d_s(u)$ is odd (even, resp.) for every $u \in V_i \setminus \{v\}$, and
- $d_s(u)$ is even (odd, resp.) for every $u \in V_{3-i}$.

Note: Remains true, even if some "contribution" from outside



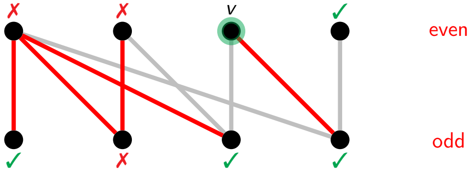
A useful lemma

Lemma

Let $s \in \{2,3\}$, and let H be a connected bipartite graph whose edges are labelled 1 or s. Consider any vertex v in any part $V_i \in \{V_1, V_2\}$ of H. We can relabel the edges of H with 1 and s so that:

- $d_s(u)$ is odd (even, resp.) for every $u \in V_i \setminus \{v\}$, and
- $d_s(u)$ is even (odd, resp.) for every $u \in V_{3-i}$.

Note: Remains true, even if some "contribution" from outside



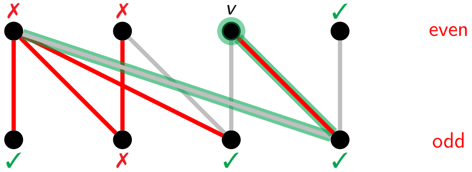
A useful lemma

Lemma

Let $s \in \{2,3\}$, and let H be a connected bipartite graph whose edges are labelled 1 or s. Consider any vertex v in any part $V_i \in \{V_1, V_2\}$ of H. We can relabel the edges of H with 1 and s so that:

- $d_s(u)$ is odd (even, resp.) for every $u \in V_i \setminus \{v\}$, and
- $d_s(u)$ is even (odd, resp.) for every $u \in V_{3-i}$.

Note: Remains true, even if some "contribution" from outside



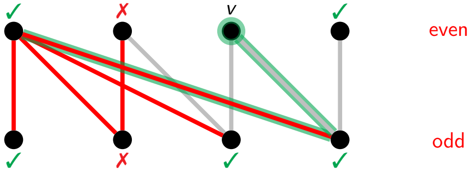
A useful lemma

Lemma

Let $s \in \{2,3\}$, and let H be a connected bipartite graph whose edges are labelled 1 or s. Consider any vertex v in any part $V_i \in \{V_1, V_2\}$ of H. We can relabel the edges of H with 1 and s so that:

- $d_s(u)$ is odd (even, resp.) for every $u \in V_i \setminus \{v\}$, and
- $d_s(u)$ is even (odd, resp.) for every $u \in V_{3-i}$.

Note: Remains true, even if some "contribution" from outside



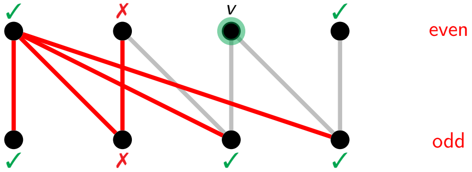
A useful lemma

Lemma

Let $s \in \{2,3\}$, and let H be a connected bipartite graph whose edges are labelled 1 or s. Consider any vertex v in any part $V_i \in \{V_1, V_2\}$ of H. We can relabel the edges of H with 1 and s so that:

- $d_s(u)$ is odd (even, resp.) for every $u \in V_i \setminus \{v\}$, and
- $d_s(u)$ is even (odd, resp.) for every $u \in V_{3-i}$.

Note: Remains true, even if some "contribution" from outside



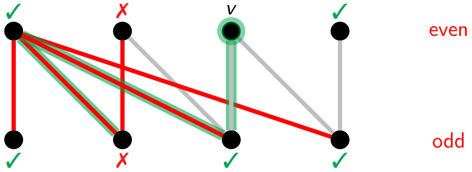
A useful lemma

Lemma

Let $s \in \{2,3\}$, and let H be a connected bipartite graph whose edges are labelled 1 or s. Consider any vertex v in any part $V_i \in \{V_1, V_2\}$ of H. We can relabel the edges of H with 1 and s so that:

- $d_s(u)$ is odd (even, resp.) for every $u \in V_i \setminus \{v\}$, and
- $d_s(u)$ is even (odd, resp.) for every $u \in V_{3-i}$.

Note: Remains true, even if some "contribution" from outside



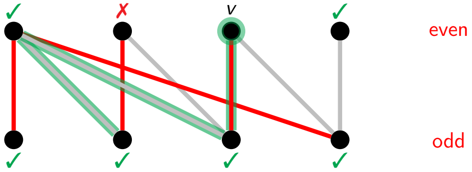
A useful lemma

Lemma

Let $s \in \{2,3\}$, and let H be a connected bipartite graph whose edges are labelled 1 or s. Consider any vertex v in any part $V_i \in \{V_1, V_2\}$ of H. We can relabel the edges of H with 1 and s so that:

- $d_s(u)$ is odd (even, resp.) for every $u \in V_i \setminus \{v\}$, and
- $d_s(u)$ is even (odd, resp.) for every $u \in V_{3-i}$.

Note: Remains true, even if some "contribution" from outside



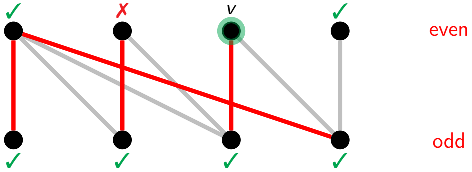
A useful lemma

Lemma

Let $s \in \{2,3\}$, and let H be a connected bipartite graph whose edges are labelled 1 or s. Consider any vertex v in any part $V_i \in \{V_1, V_2\}$ of H. We can relabel the edges of H with 1 and s so that:

- $d_s(u)$ is odd (even, resp.) for every $u \in V_i \setminus \{v\}$, and
- $d_s(u)$ is even (odd, resp.) for every $u \in V_{3-i}$.

Note: Remains true, even if some “contribution” from outside



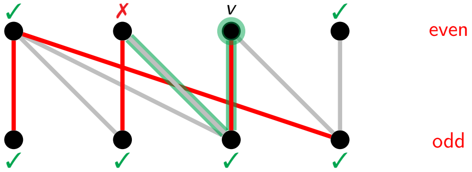
A useful lemma

Lemma

Let $s \in \{2,3\}$, and let H be a connected bipartite graph whose edges are labelled 1 or s. Consider any vertex v in any part $V_i \in \{V_1, V_2\}$ of H. We can relabel the edges of H with 1 and s so that:

- $d_s(u)$ is odd (even, resp.) for every $u \in V_i \setminus \{v\}$, and
- $d_s(u)$ is even (odd, resp.) for every $u \in V_{3-i}$.

Note: Remains true, even if some "contribution" from outside



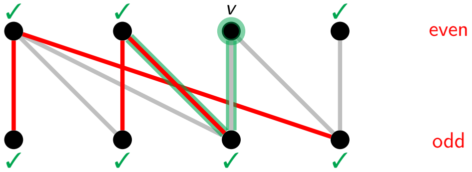
A useful lemma

Lemma

Let $s \in \{2,3\}$, and let H be a connected bipartite graph whose edges are labelled 1 or s. Consider any vertex v in any part $V_i \in \{V_1, V_2\}$ of H. We can relabel the edges of H with 1 and s so that:

- $d_s(u)$ is odd (even, resp.) for every $u \in V_i \setminus \{v\}$, and
- $d_s(u)$ is even (odd, resp.) for every $u \in V_{3-i}$.

Note: Remains true, even if some "contribution" from outside



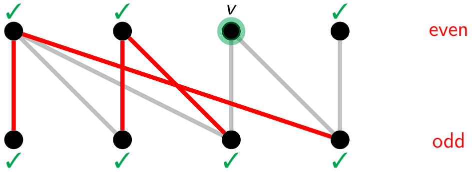
A useful lemma

Lemma

Let $s \in \{2,3\}$, and let H be a connected bipartite graph whose edges are labelled 1 or s. Consider any vertex v in any part $V_i \in \{V_1, V_2\}$ of H. We can relabel the edges of H with 1 and s so that:

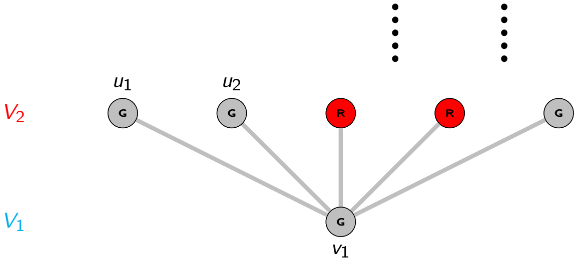
- $d_s(u)$ is odd (even, resp.) for every $u \in V_i \setminus \{v\}$, and
- $d_s(u)$ is even (odd, resp.) for every $u \in V_{3-i}$.

Note: Remains true, even if some "contribution" from outside



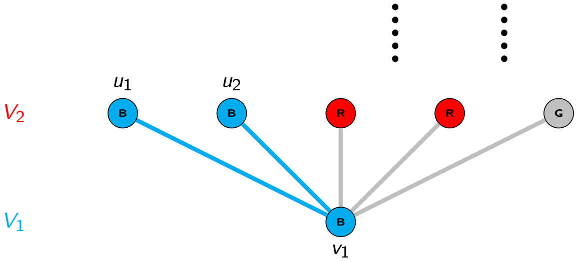
Case 1: H has either 1) a 1-mono $v_1 \in V_1$ with two 1-mono degre-1 neighbours $u_1, u_2 \in V_2$, or 2) a 3-mono $v \in V_1$

First situation:



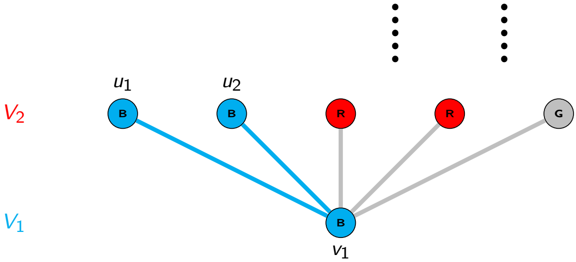
Case 1: H has either 1) a 1-mono $v_1 \in V_1$ with two 1-mono degre-1 neighbours $u_1, u_2 \in V_2$, or 2) a 3-mono $v \in V_1$

First situation:



Case 1: H has either 1) a 1-mono $v_1 \in V_1$ with two 1-mono degre-1 neighbours $u_1, u_2 \in V_2$, or 2) a 3-mono $v \in V_1$

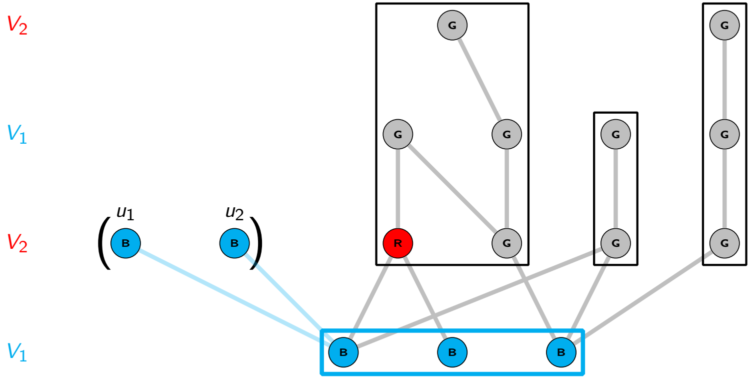
First situation:



... and then 2nd situation, keeping in mind that the only 3-mono in V_2 are u_1, u_2

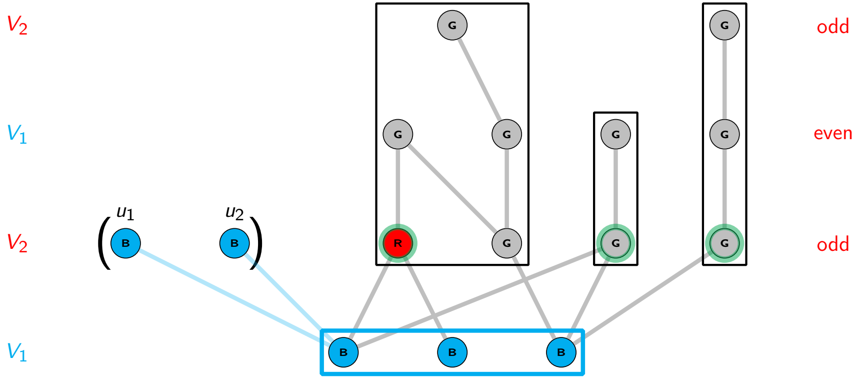
Case 1: H has either 1) a 1-mono $v_1 \in V_1$ with two 1-mono degre-1 neighbours $u_1, u_2 \in V_2$, or 2) a 3-mono $v \in V_1$

Second situation:



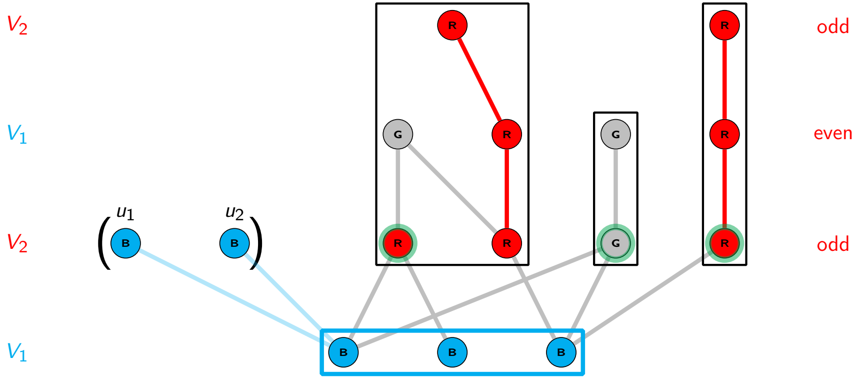
Case 1: H has either 1) a 1-mono $v_1 \in V_1$ with two 1-mono degre-1 neighbours $u_1, u_2 \in V_2$, or 2) a 3-mono $v \in V_1$

Second situation:



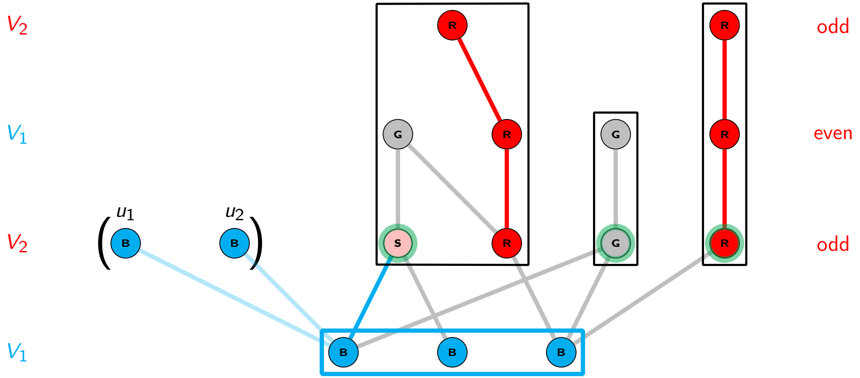
Case 1: H has either 1) a 1-mono $v_1 \in V_1$ with two 1-mono degre-1 neighbours $u_1, u_2 \in V_2$, or 2) a 3-mono $v \in V_1$

Second situation:



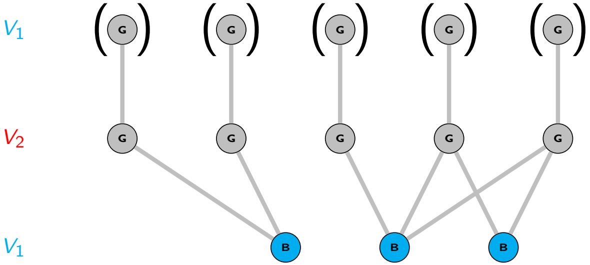
Case 1: H has either 1) a 1-mono $v_1 \in V_1$ with two 1-mono degre-1 neighbours $u_1, u_2 \in V_2$, or 2) a 3-mono $v \in V_1$

Second situation:



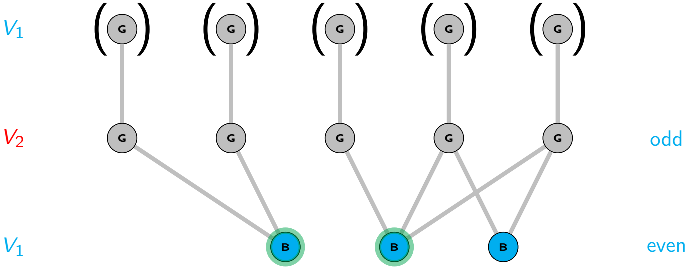
Case 1: H has either 1) a 1-mono $v_1 \in V_1$ with two 1-mono degre-1 neighbours $u_1, u_2 \in V_2$, or 2) a 3-mono $v \in V_1$

Second situation:



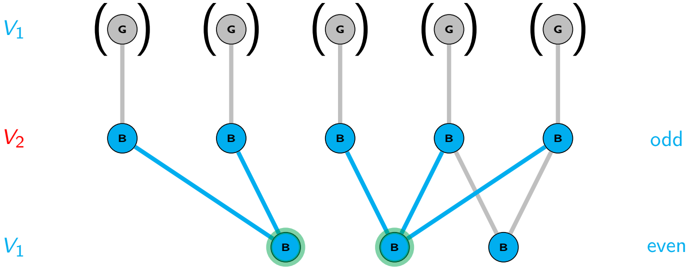
Case 1: H has either 1) a 1-mono $v_1 \in V_1$ with two 1-mono degre-1 neighbours $u_1, u_2 \in V_2$, or 2) a 3-mono $v \in V_1$

Second situation:



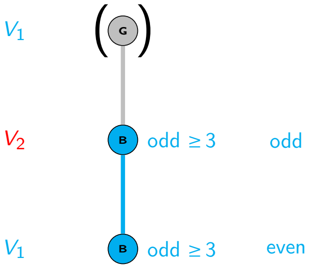
Case 1: H has either 1) a 1-mono $v_1 \in V_1$ with two 1-mono degre-1 neighbours $u_1, u_2 \in V_2$, or 2) a 3-mono $v \in V_1$

Second situation:



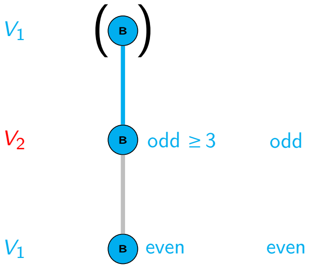
Case 1: H has either 1) a 1-mono $v_1 \in V_1$ with two 1-mono degre-1 neighbours $u_1, u_2 \in V_2$, or 2) a 3-mono $v \in V_1$

Second situation:



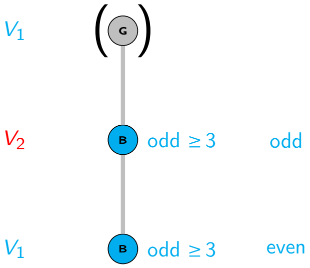
Case 1: H has either 1) a 1-mono $v_1 \in V_1$ with two 1-mono degre-1 neighbours $u_1, u_2 \in V_2$, or 2) a 3-mono $v \in V_1$

Second situation:



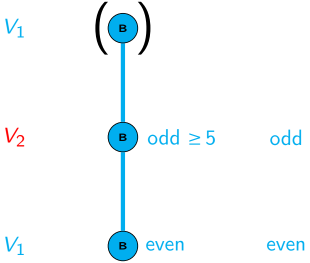
Case 1: H has either 1) a 1-mono $v_1 \in V_1$ with two 1-mono degre-1 neighbours $u_1, u_2 \in V_2$, or 2) a 3-mono $v \in V_1$

Second situation:

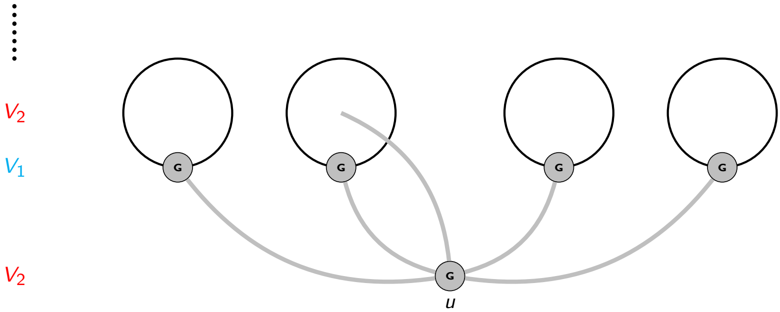


Case 1: H has either 1) a 1-mono $v_1 \in V_1$ with two 1-mono degre-1 neighbours $u_1, u_2 \in V_2$, or 2) a 3-mono $v \in V_1$

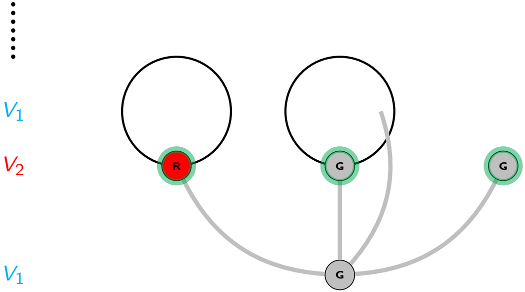
Second situation:



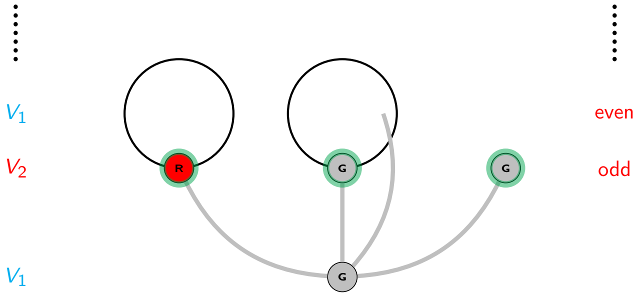
Case 2: H contains a 1-mono $u \in V_2$ with at least two neighbours in H



Looking closer at components of $H - u$



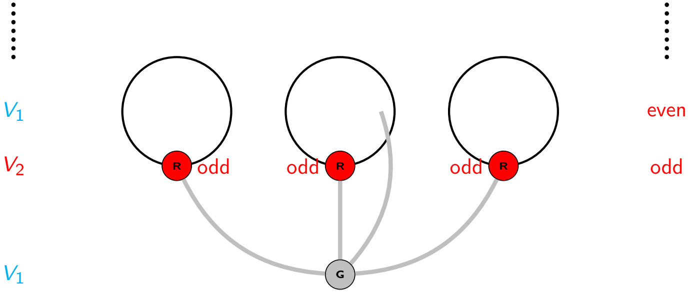
Looking closer at components of $H - u$



Nice components

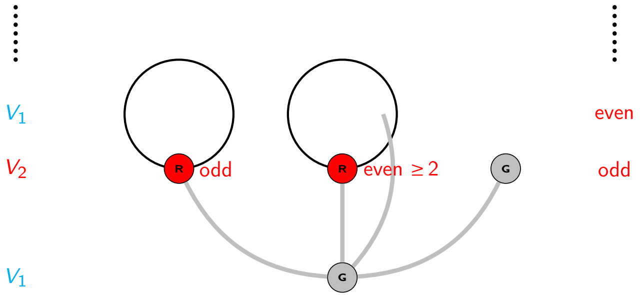
Nice component: no conflict, or at least two neighbours with even 2-degree, or only one neighbour with even 2-degree at least 2

$\Rightarrow$ **can make sure no conflict in the component!**



Nice components

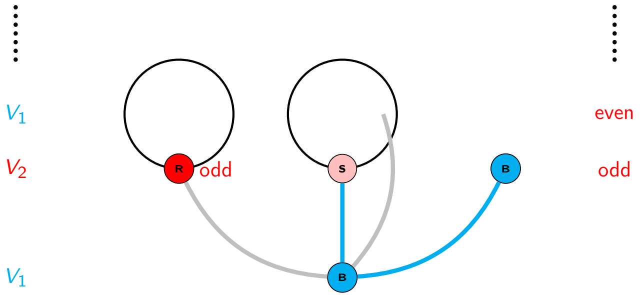
Nice component: no conflict, or at least two neighbours with even 2-degree, or only one neighbour with even 2-degree at least 2
$\Rightarrow$ **can make sure no conflict in the component!**



Nice components

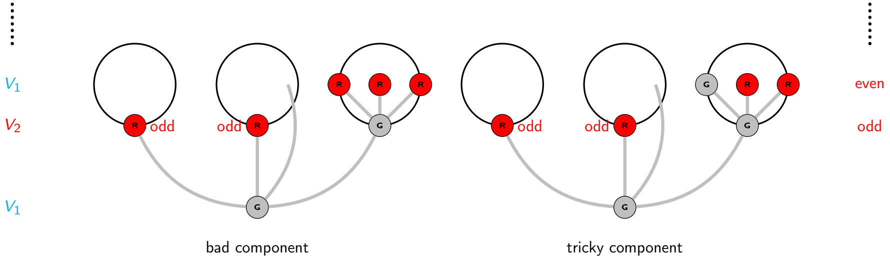
Nice component: no conflict, or at least two neighbours with even 2-degree, or only one neighbour with even 2-degree at least 2

⇒ **can make sure no conflict in the component!**

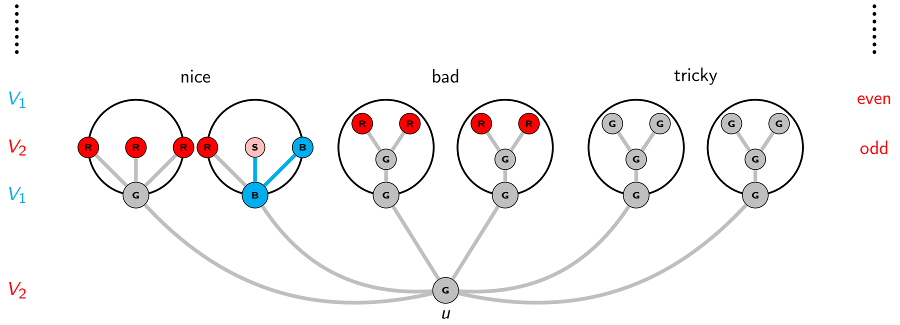


Bad and tricky components

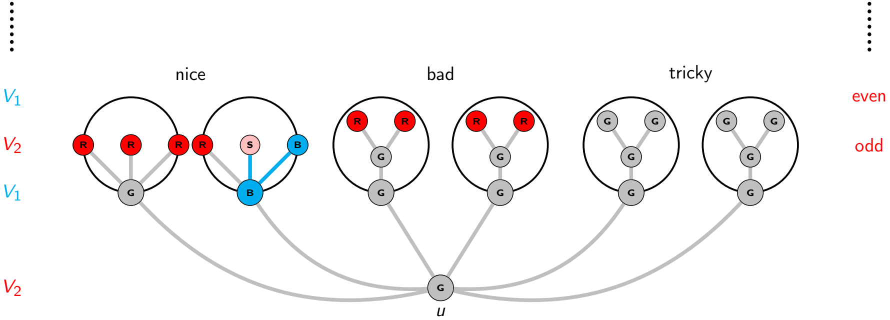
Bad component: exactly one neighbour with even 2-degree, being 1-mono
Tricky component: that 1-mono neighbour is adjacent to a 1-mono neighbour



Global picture



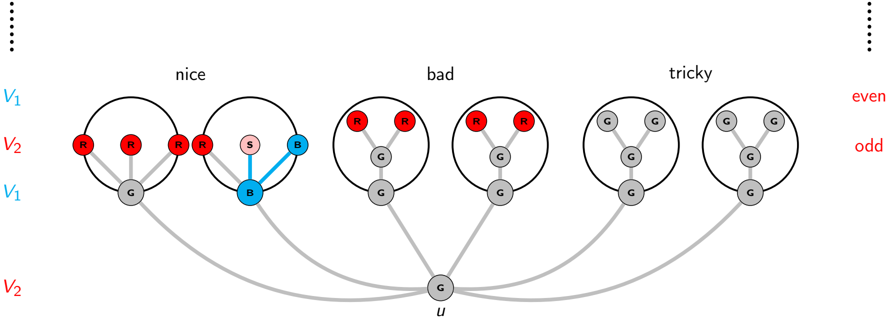
Global picture



Some terminology:

- N_n: number of nice components

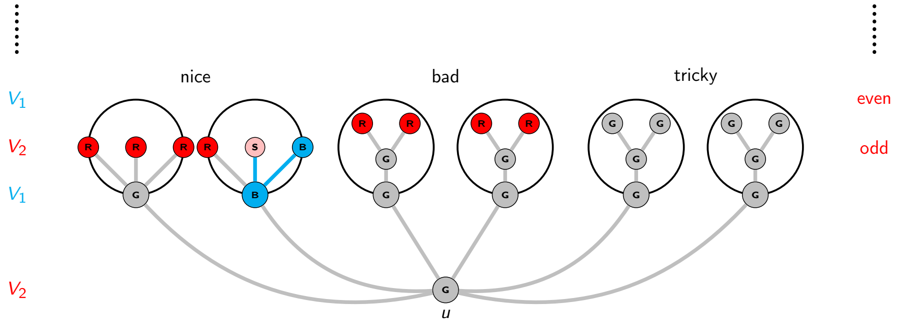
Global picture



Some terminology:

- N_n: number of nice components
- N_b: number of bad components

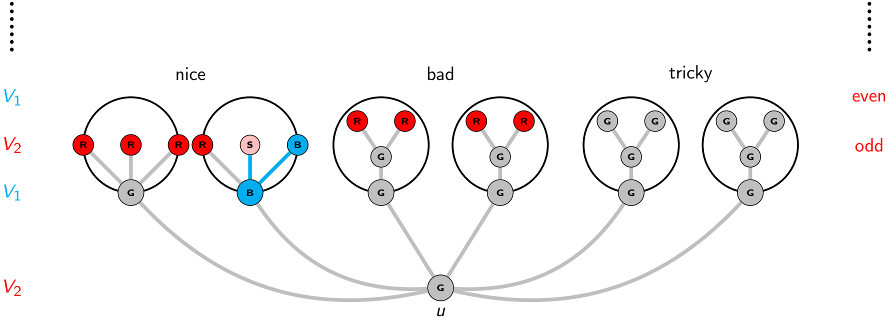
Global picture



Some terminology:

- N_n: number of nice components
- N_b: number of bad components
- N_t: number of tricky components

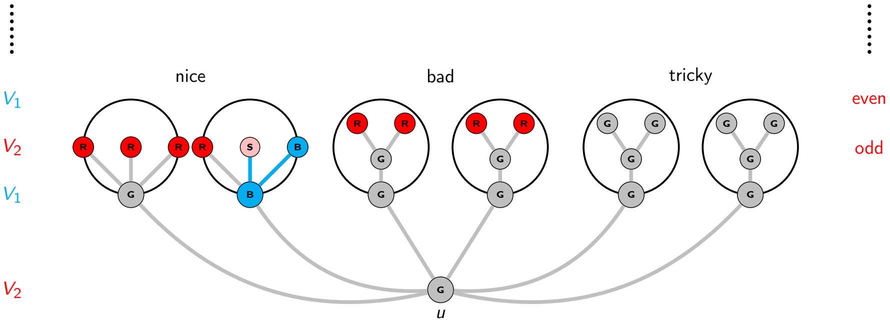
Global picture



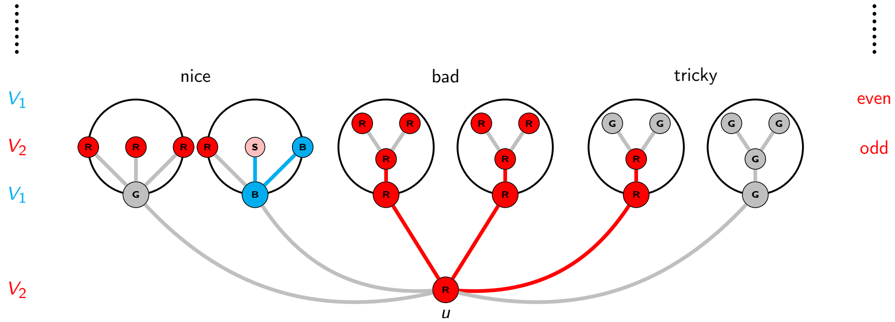
Some terminology:

- N_n: number of nice components
- N_b: number of bad components
- N_t: number of tricky components
- N_{an}: number of neighbours with 2-degree 0 in nice components ($N_{an} \geq N_n$)

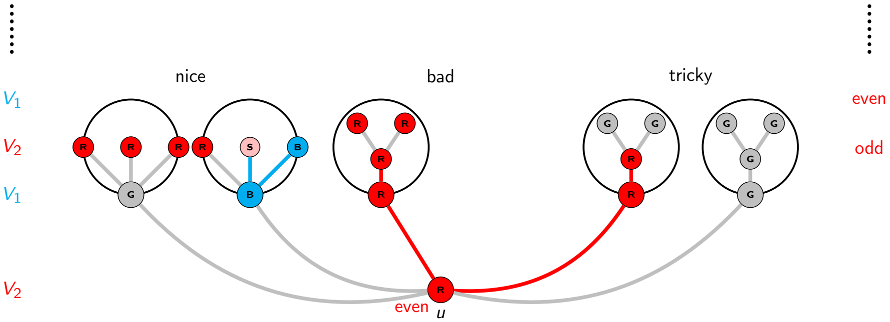
First case: $N_t > 0$



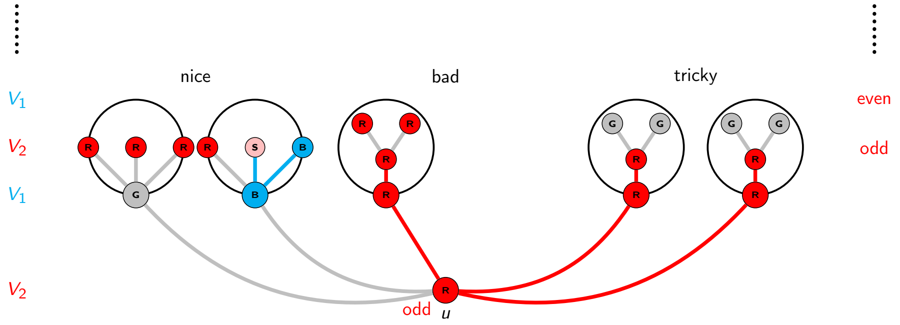
First case: $N_t > 0$



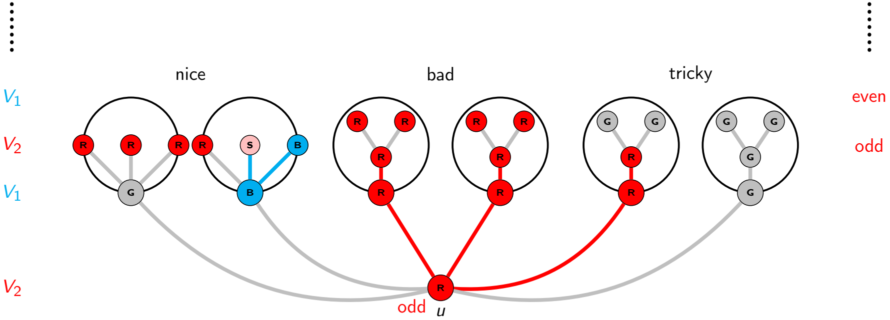
First case: $N_t > 0$



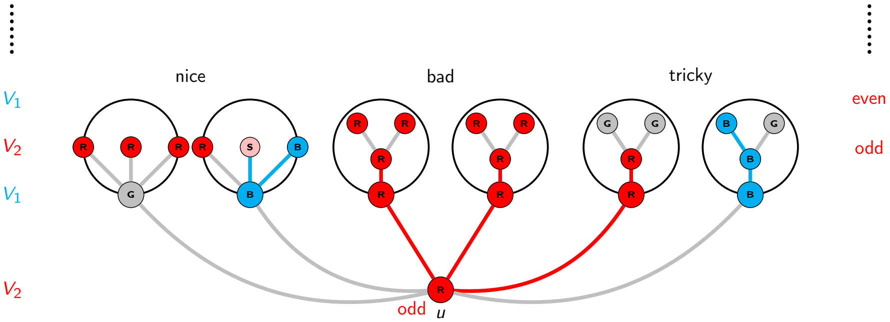
First case: $N_t > 0$



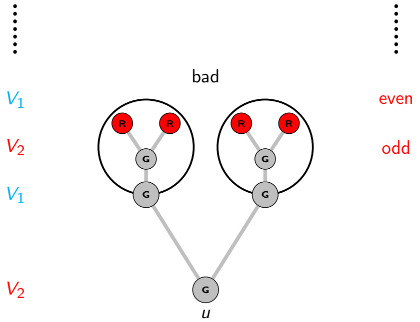
First case: $N_t > 0$



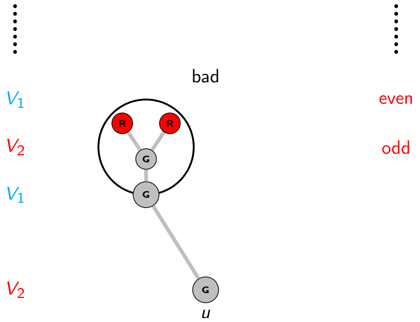
First case: $N_t > 0$



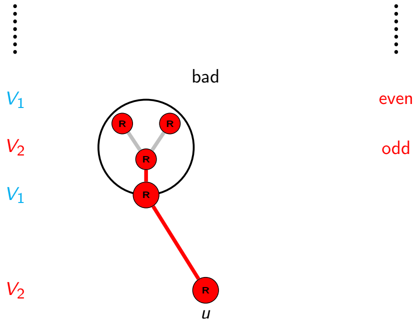
Second case: $N_{an} = N_n = 0$



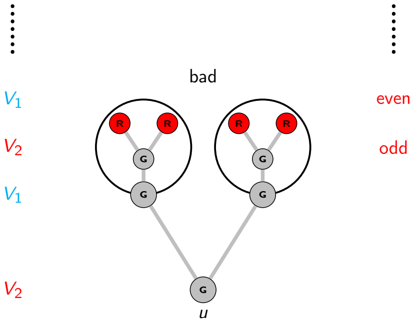
Second case: $N_{an} = N_n = 0$



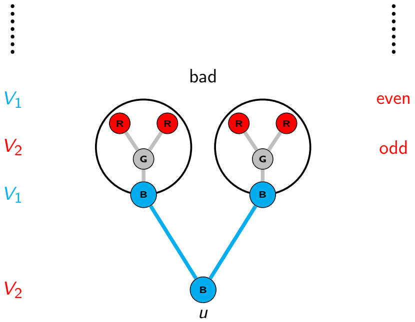
Second case: $N_{an} = N_n = 0$



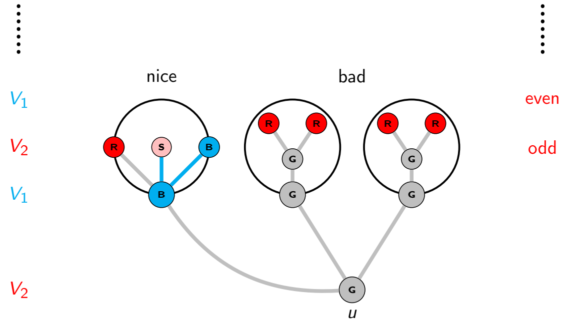
Second case: $N_{an} = N_n = 0$



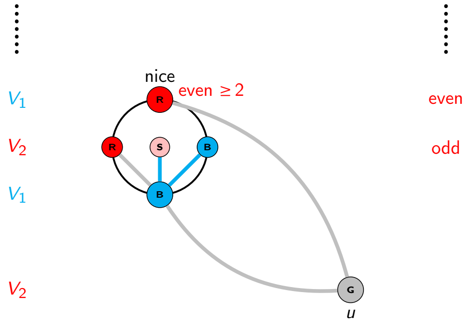
Second case: $N_{an} = N_n = 0$



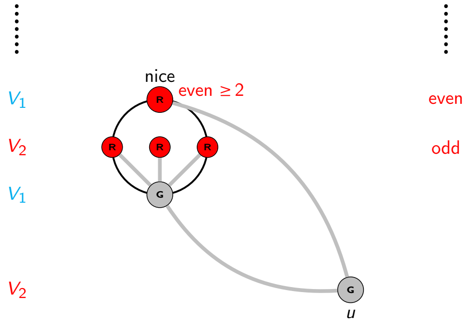
Third case: $N_{an} = 1$



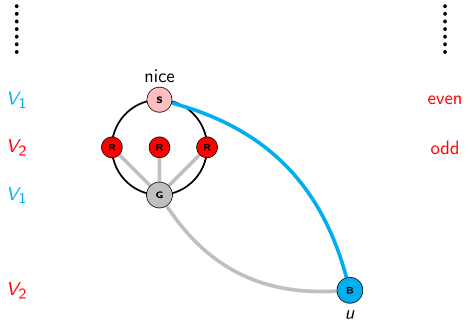
Third case: $N_{an} = 1$



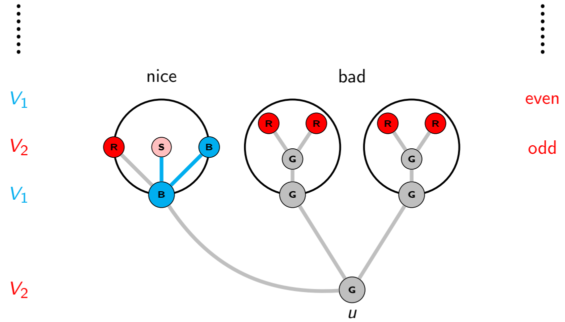
Third case: $N_{an} = 1$



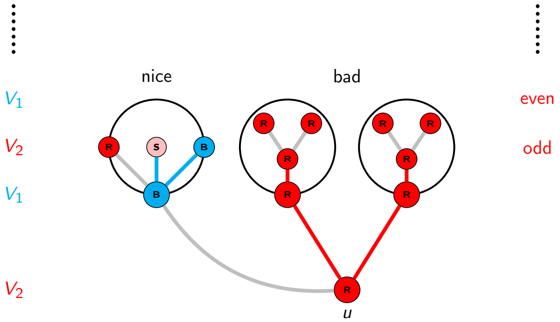
Third case: $N_{an} = 1$



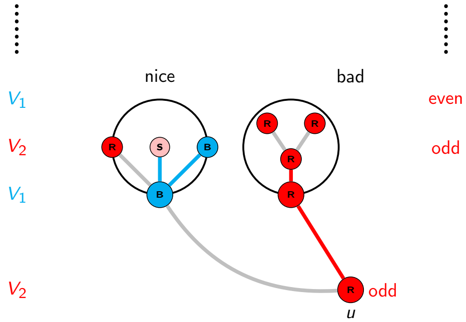
Third case: $N_{an} = 1$



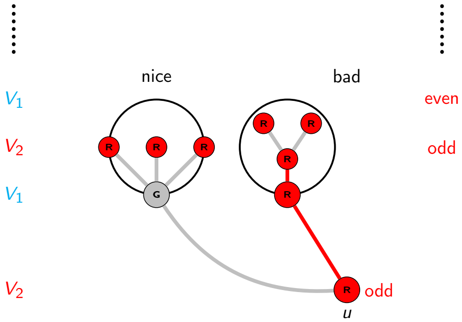
Third case: $N_{an} = 1$



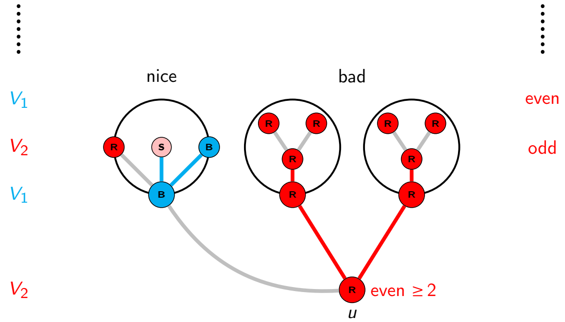
Third case: $N_{an} = 1$



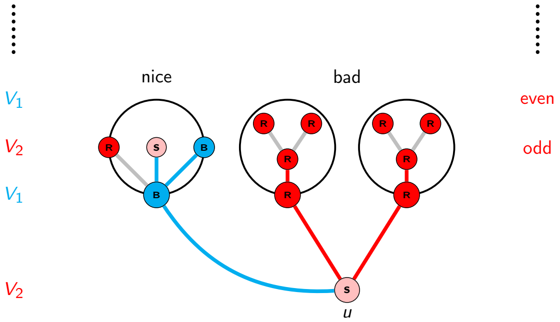
Third case: $N_{an} = 1$



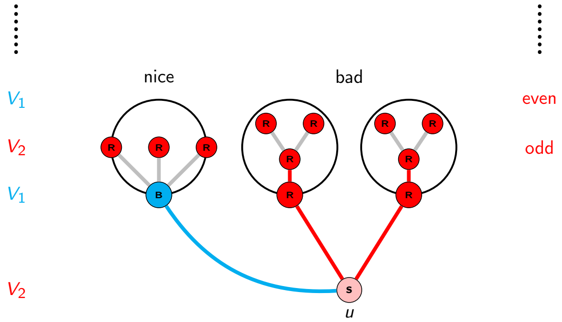
Third case: $N_{an} = 1$



Third case: $N_{an} = 1$

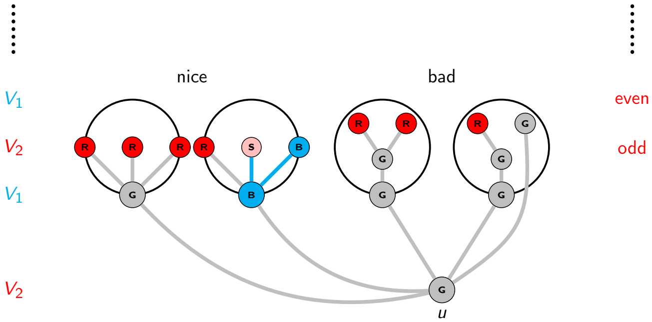


Third case: $N_{an} = 1$



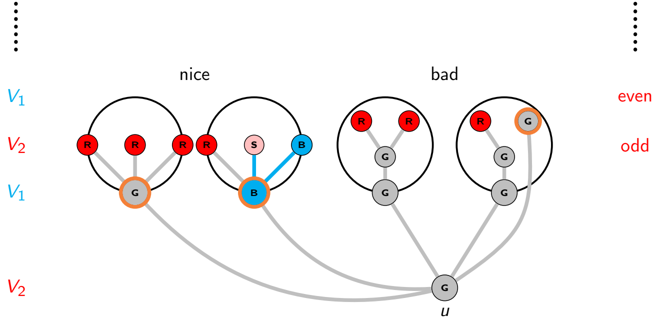
Fourth (and last) case: $N_{an} \geq 2$

$A = \{a_1, \ldots, a_r\}$: neighbours with 2-degree 0 not "main neighbour" in a bad comp.



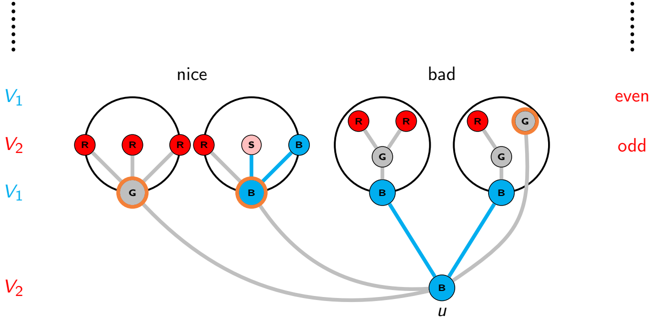
Fourth (and last) case: $N_{an} \geq 2$

$A = \{a_1, \ldots, a_r\}$: neighbours with 2-degree 0 not "main neighbour" in a bad comp.



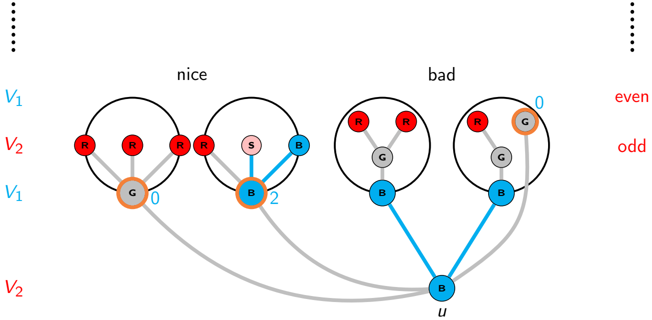
Fourth (and last) case: $N_{an} \geq 2$

$A = \{a_1, \ldots, a_r\}$: neighbours with 2-degree 0 not "main neighbour" in a bad comp.



Fourth (and last) case: $N_{an} \geq 2$

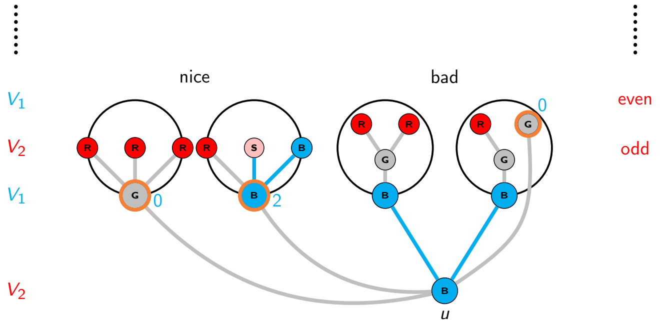
$A = \{a_1, \ldots, a_r\}$: neighbours with 2-degree 0 not "main neighbour" in a bad comp.



For every $i \in \{1, \ldots, r\}$, set $n_i := d_3(a_i)$

Fourth (and last) case: $N_{an} \geq 2$

$A = \{a_1, \dots, a_r\}$: neighbours with 2-degree 0 not "main neighbour" in a bad comp.

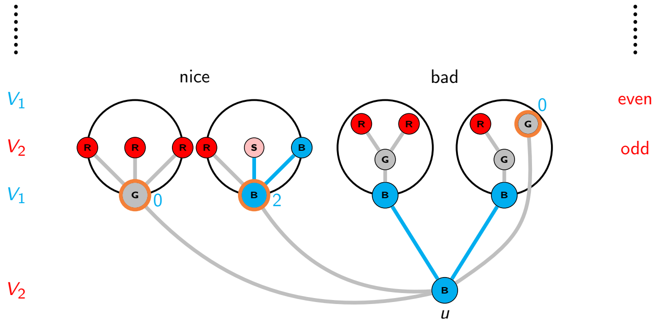


For every $i \in \{1, \dots, r\}$, set $n_i := d_3(a_i)$

Goal: Relabel some ua_i's with 3 so that u is not in conflict with the a_i's

Fourth (and last) case: $N_{an} \geq 2$

$A = \{a_1, \ldots, a_r\}$: neighbours with 2-degree 0 not "main neighbour" in a bad comp.



For every $i \in \{1, \ldots, r\}$, set $n_i := d_3(a_i)$

Goal: Relabel some ua_i's with 3 so that u is not in conflict with the a_i's
$\Rightarrow$ possible because $N_{an} \geq 2$

Polynomial representation

- For every $i \in \{1, \ldots, r\}$, let X_i be a variable taking value in $\{0, 1\}$
- $X_i = 0$ means label 1 on ua_i, while $X_i = 1$ means label 3 on ua_i

Polynomial representation

- For every $i \in \{1, \ldots, r\}$, let X_i be a variable taking value in $\{0, 1\}$
- $X_i = 0$ means label 1 on ua_i, while $X_i = 1$ means label 3 on ua_i
- Model the constraints by the following polynomial:

$$P(X_1, \ldots, X_r) = \prod_{i=1}^{r} \left(\sum_{\substack{j=1 \\ j \neq i}}^{r} X_i + N_b - n_i \right)$$

Polynomial representation

- For every $i \in \{1, \ldots, r\}$, let X_i be a variable taking value in $\{0,1\}$
- $X_i = 0$ means label 1 on ua_i, while $X_i = 1$ means label 3 on ua_i
- Model the constraints by the following polynomial:

$$P(X_1, \ldots, X_r) = \prod_{i=1}^{r} \left(\sum_{\substack{j=1 \\ j \neq i}}^{r} X_i + N_b - n_i \right)$$

- For $x_1, \ldots, x_r \in \{0,1\}$, have $P(x_1, \ldots, x_r) \neq 0$ iff none of the mentioned conflicts

Polynomial representation

- For every $i \in \{1,\ldots,r\}$, let X_i be a variable taking value in $\{0,1\}$
- $X_i = 0$ means label 1 on ua_i, while $X_i = 1$ means label 3 on ua_i
- Model the constraints by the following polynomial:

$$P(X_1,\ldots,X_r) = \prod_{i=1}^{r}\left(\sum_{\substack{j=1\\ j\neq i}}^{r} X_i + N_b - n_i\right)$$

- For $x_1,\ldots,x_r \in \{0,1\}$, have $P(x_1,\ldots,x_r) \neq 0$ iff none of the mentioned conflicts

Combinatorial Nullstellensatz (Alon, 1999)

Let $\mathbb{F}$ be an arbitrary field, and let $f = f(x_1,\ldots,x_n)$ be a polynomial in $\mathbb{F}[x_1,\ldots,x_n]$. Suppose the total degree of f is $\sum_{i=1}^{n} t_i$, where each t_i is a non-negative integer, and suppose the coefficient of $\prod_{i=1}^{n} x_i^{t_i}$ is non-zero. If $S_1,\ldots,S_n$ are subsets of $\mathbb{F}$ with $|S_i| > t_i$, then there are $s_1 \in S_1, s_2 \in S_2,\ldots,s_n \in S_n$ so that $f(s_1,\ldots,s_n) \neq 0$.

Polynomial representation

- For every $i \in \{1, \ldots, r\}$, let X_i be a variable taking value in $\{0, 1\}$
- $X_i = 0$ means label 1 on ua_i, while $X_i = 1$ means label 3 on ua_i
- Model the constraints by the following polynomial:

$$P(X_1, \ldots, X_r) = \prod_{i=1}^{r} \left(\sum_{\substack{j=1 \\ j \neq i}}^{r} X_i + N_b - n_i \right)$$

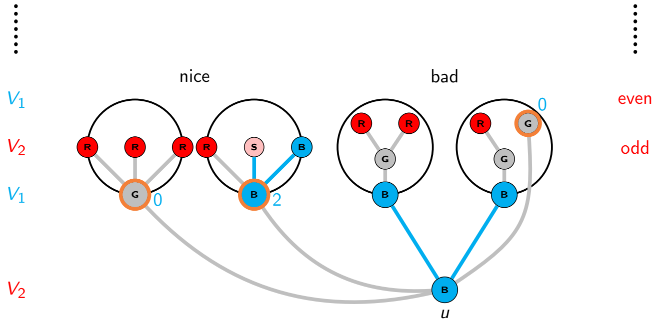
- For $x_1, \ldots, x_r \in \{0, 1\}$, have $P(x_1, \ldots, x_r) \neq 0$ iff none of the mentioned conflicts

Combinatorial Nullstellensatz (Alon, 1999)

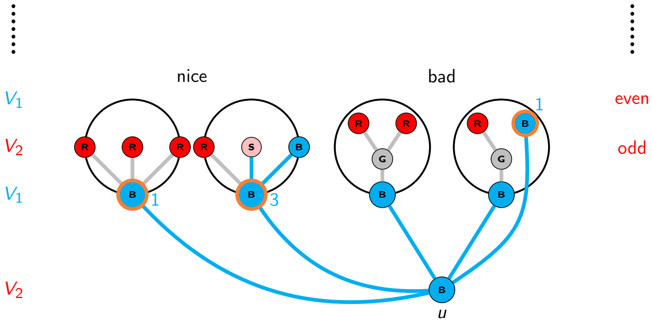
Let $\mathbb{F}$ be an arbitrary field, and let $f = f(x_1, \ldots, x_n)$ be a polynomial in $\mathbb{F}[x_1, \ldots, x_n]$. Suppose the total degree of f is $\sum_{i=1}^{n} t_i$, where each t_i is a non-negative integer, and suppose the coefficient of $\prod_{i=1}^{n} x_i^{t_i}$ is non-zero. If $S_1, \ldots, S_n$ are subsets of $\mathbb{F}$ with $|S_i| > t_i$, then there are $s_1 \in S_1, s_2 \in S_2, \ldots, s_n \in S_n$ so that $f(s_1, \ldots, s_n) \neq 0$.

- Here, just consider the monomial $\prod_{i=1}^{r} X_i$ $\Rightarrow$ the desired x_i's exist!

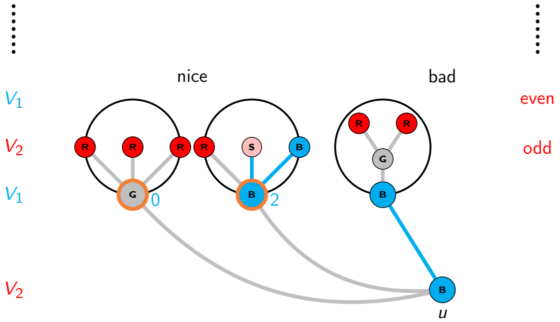
Fourth (and last) case: $N_{an} \geq 2$



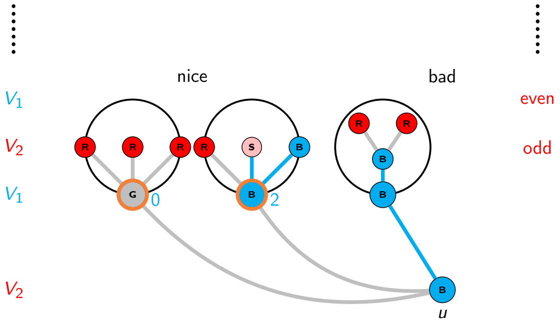
Fourth (and last) case: $N_{an} \geq 2$



Fourth (and last) case: $N_{an} \geq 2$

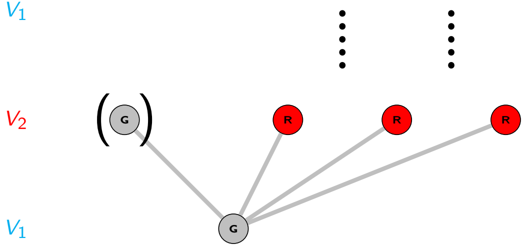


Fourth (and last) case: $N_{an} \geq 2$



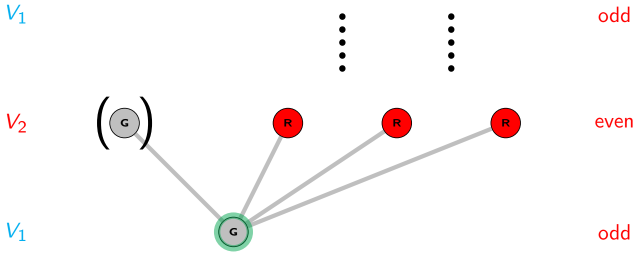
Case 3: None of Cases 1 and 2 applies

1. H contains either:
   - a 1-mono $v_1 \in V_1$ with two 1-mono degre-1 neighbours $u_1, u_2 \in V_2$, or
   - a 3-mono $v \in V_1$
2. H contains a 1-mono $u \in V_2$ with at least two neighbours in H



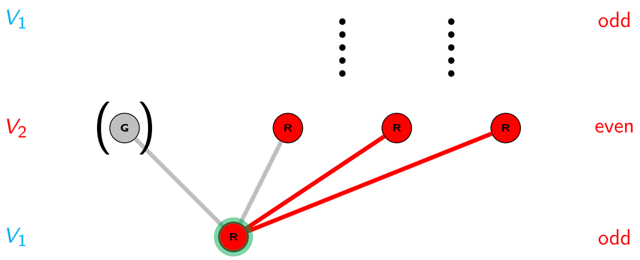
Case 3: None of Cases 1 and 2 applies

1. H contains either:
   - a 1-mono $v_1 \in V_1$ with two 1-mono degre-1 neighbours $u_1, u_2 \in V_2$, or
   - a 3-mono $v \in V_1$
2. H contains a 1-mono $u \in V_2$ with at least two neighbours in H



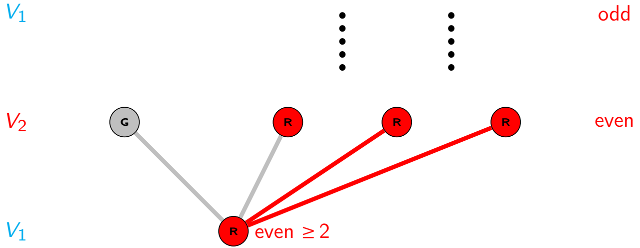
Case 3: None of Cases 1 and 2 applies

1. H contains either:
   - a 1-mono $v_1 \in V_1$ with two 1-mono degre-1 neighbours $u_1, u_2 \in V_2$, or
   - a 3-mono $v \in V_1$
2. H contains a 1-mono $u \in V_2$ with at least two neighbours in H



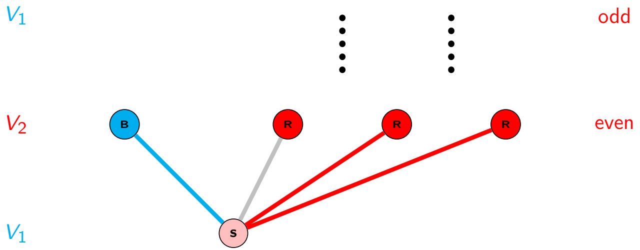
Case 3: None of Cases 1 and 2 applies

1. H contains either:
   - a 1-mono $v_1 \in V_1$ with two 1-mono degre-1 neighbours $u_1, u_2 \in V_2$, or
   - a 3-mono $v \in V_1$
2. H contains a 1-mono $u \in V_2$ with at least two neighbours in H



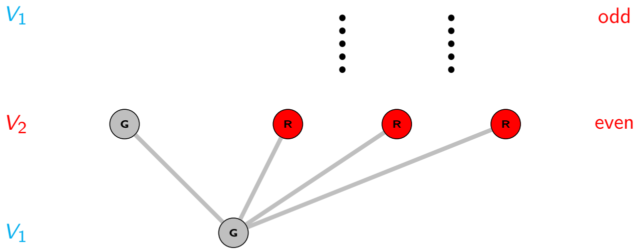
Case 3: None of Cases 1 and 2 applies

1. H contains either:
   - a 1-mono $v_1 \in V_1$ with two 1-mono degre-1 neighbours $u_1, u_2 \in V_2$, or
   - a 3-mono $v \in V_1$
2. H contains a 1-mono $u \in V_2$ with at least two neighbours in H



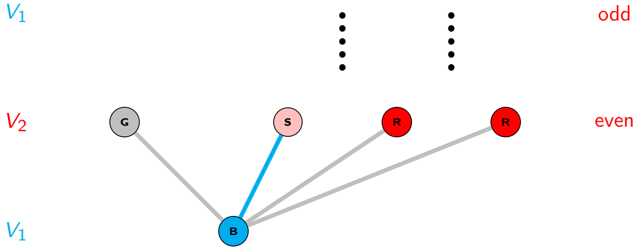
Case 3: None of Cases 1 and 2 applies

1. H contains either:
   - a 1-mono $v_1 \in V_1$ with two 1-mono degre-1 neighbours $u_1, u_2 \in V_2$, or
   - a 3-mono $v \in V_1$
2. H contains a 1-mono $u \in V_2$ with at least two neighbours in H



Case 3: None of Cases 1 and 2 applies

1. H contains either:
   - a 1-mono $v_1 \in V_1$ with two 1-mono degre-1 neighbours $u_1, u_2 \in V_2$, or
   - a 3-mono $v \in V_1$
2. H contains a 1-mono $u \in V_2$ with at least two neighbours in H



End of the proof, phew...

☺ ☺ ☺ ☺ ☺

Thank you for your attention!