Strong edge-coloring of $(3, \Delta)$-bipartite graphs

Julien Bensmail[a], Aurélie Lagoutte[a] and Petru Valicov[b]

a. LIP – ENS de Lyon – France
b. LIF – Université Aix-Marseille – France

LIRIS
December 12th, 2014

Strong edge-coloring

G: undirected simple graph
c: edge-coloring of G

Definition: *strong edge-coloring*

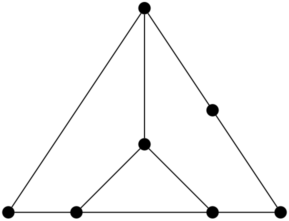
We call c *strong* if every two edges at distance at most 2 in G are assigned distinct colors by c.

Strong edge-coloring

G: undirected simple graph
c: edge-coloring of G

Definition: *strong edge-coloring*

We call c *strong* if every two edges at distance at most 2 in G are assigned distinct colors by c.

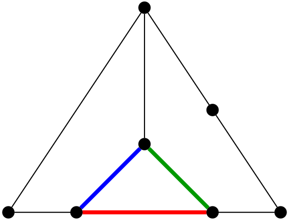


Strong edge-coloring

G: undirected simple graph
c: edge-coloring of G

Definition: *strong edge-coloring*

We call c *strong* if every two edges at distance at most 2 in G are assigned distinct colors by c.

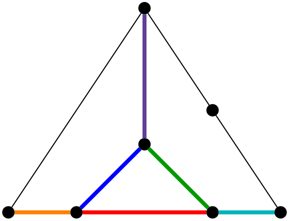


Strong edge-coloring

G: undirected simple graph
c: edge-coloring of G

Definition: *strong edge-coloring*

We call c *strong* if every two edges at distance at most 2 in G are assigned distinct colors by c.

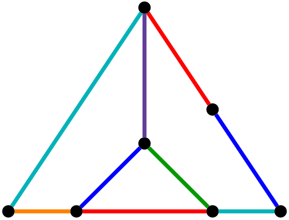


Strong edge-coloring

G: undirected simple graph
c: edge-coloring of G

Definition: *strong edge-coloring*

We call c *strong* if every two edges at distance at most 2 in G are assigned distinct colors by c.

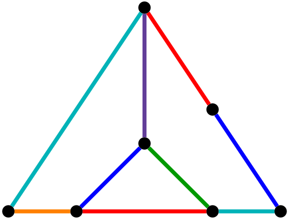


Strong edge-coloring

G: undirected simple graph
c: edge-coloring of G

Definition: *strong edge-coloring*

We call c *strong* if every two edges at distance at most 2 in G are assigned distinct colors by c.



Equivalently:

- edge-partition giving *induced* matchings
- proper vertex-coloring of $L(G)^2$

Strong chromatic index

Δ: maximum degree of an explicit graph

Definition: *strong chromatic index*

The least number of colors in a strong edge-coloring of G is the *strong chromatic index* of G, denoted $\chi'_s(G)$.

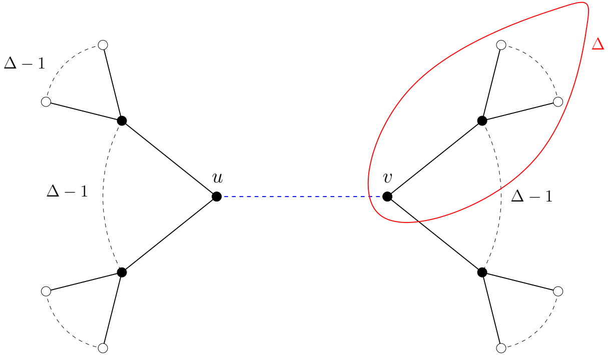
Strong chromatic index

Δ: maximum degree of an explicit graph

Definition: *strong chromatic index*

The least number of colors in a strong edge-coloring of G is the *strong chromatic index* of G, denoted $\chi'_s(G)$.

Brooks-like argument: $\chi'_s(G) \leq 2\Delta^2 - 2\Delta + 1$ ($\approx 2\Delta^2$)



On the Brooks-like upper bound on χ'_s

optimality of $2\Delta^2$?

Theorem [Molloy, Reed – 1997]

If Δ is large enough, then $\chi'_s(G) \leq 1.998\Delta^2$.

On the Brooks-like upper bound on χ'_s

optimality of $2\Delta^2$?

Theorem [Molloy, Reed – 1997]

If Δ is large enough, then $\chi'_s(G) \leq 1.998\Delta^2$.

What would be a "worst graph"? C_5^Δ:

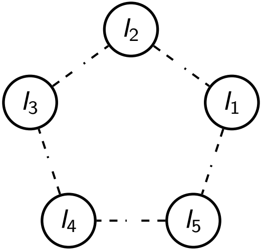
On the Brooks-like upper bound on χ'_s

optimality of $2\Delta^2$?

Theorem [Molloy, Reed – 1997]

If Δ is large enough, then $\chi'_s(G) \leq 1.998\Delta^2$.

What would be a "worst graph"? C_5^Δ:



- every I_j is an independent set,
- two "adjacent" I_j's are complete to each other,
- if $\Delta = 2k$, then $|I_j| = k$,
- if $\Delta = 2k + 1$, then $|I_1| = |I_2| = |I_3| = k$,

  and $|I_4| = |I_5| = k + 1$.

Erdős and Nešetřil's conjecture

Conjecture [Erdős, Nešetřil – 1985]

We have $\chi'_s(G) \leq \begin{cases} \frac{5}{4}\Delta^2 \text{ for } \Delta \text{ even, and} \\ \frac{1}{4}(5\Delta^2 - 2\Delta + 1) \text{ otherwise.} \end{cases}$

Erdős and Nešetřil's conjecture

Conjecture [Erdős, Nešetřil – 1985]

We have $\chi'_s(G) \leq \begin{cases} \frac{5}{4}\Delta^2 \text{ for } \Delta \text{ even, and} \\ \frac{1}{4}(5\Delta^2 - 2\Delta + 1) \text{ otherwise.} \end{cases}$

Facts:

- reached for C_5^Δ's only [Chung, Gyárfás, Tuza, Trotter – 1990]

Erdős and Nešetřil's conjecture

Conjecture [Erdős, Nešetřil – 1985]

We have $\chi'_s(G) \leq \begin{cases} \frac{5}{4}\Delta^2 \text{ for } \Delta \text{ even, and} \\ \frac{1}{4}(5\Delta^2 - 2\Delta + 1) \text{ otherwise.} \end{cases}$

Facts:

- reached for C_5^Δ's only [Chung, Gyárfás, Tuza, Trotter – 1990]
- verified for $\Delta = 3$ [Andersen – 1992]

Erdős and Nešetřil's conjecture

Conjecture [Erdős, Nešetřil – 1985]

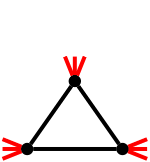
We have $\chi'_s(G) \leq \begin{cases} \frac{5}{4}\Delta^2 \text{ for } \Delta \text{ even, and} \\ \frac{1}{4}(5\Delta^2 - 2\Delta + 1) \text{ otherwise.} \end{cases}$

Facts:

- reached for C_5^Δ's only [Chung, Gyárfás, Tuza, Trotter – 1990]
- verified for $\Delta = 3$ [Andersen – 1992]
- for $\Delta = 4$, we know $\chi'_s(G) \leq 22$ [Cranston – 2006]

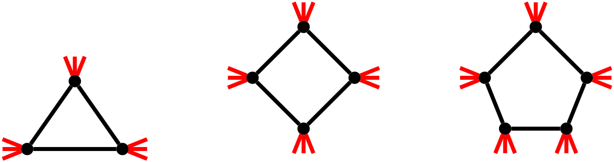
Beyond Erdős and Nešetřil's construction

Less dependencies for graphs with no small cycles



Beyond Erdős and Nešetřil's construction

Less dependencies for graphs with no small cycles

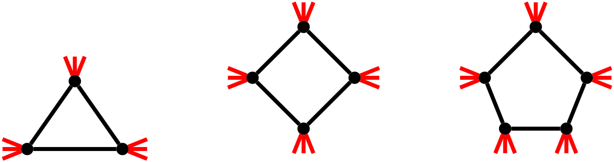


Theorem [Mahdian – 2000]

If G is C_4-free, then $\chi'_s(G) \leq (2 + o(1))\frac{\Delta^2}{\ln \Delta}$.

Beyond Erdős and Nešetřil's construction

Less dependencies for graphs with no small cycles



Theorem [Mahdian – 2000]

If G is C_4-free, then $\chi'_s(G) \leq (2 + o(1))\frac{\Delta^2}{\ln \Delta}$.

What for C_3- and C_5-free graphs?

What for bipartite graphs?

Bipartite graphs are C_3 and C_5-free...

Conjecture [Faudree, Gyárfás, Schelp, Tuza – 1990]

If G is bipartite, then $\chi'_s(G) \leq \Delta^2$.

Reached e.g. for any complete bipartite graph $K_{a,a}$

What for bipartite graphs?

Bipartite graphs are C_3 and C_5-free...

Conjecture [Faudree, Gyárfás, Schelp, Tuza – 1990]

If G is bipartite, then $\chi'_s(G) \leq \Delta^2$.

Reached e.g. for any complete bipartite graph $K_{a,a}$

$G = (A, B, E)$: bipartite graph with bipartition A and B
(Δ_A, Δ_B)-bipartite graph: A and B have maximum degree Δ_A and Δ_B, resp.

Conjecture [Brualdi, Quinn Massey – 1993]

If G is (Δ_A, Δ_B)-bipartite, then $\chi'_s(G) \leq \Delta_A \Delta_B$.

Refined conjecture for bipartite graphs

Conjecture [Brualdi, Quinn Massey – 1993]

If G is (Δ_A, Δ_B)-bipartite, then $\chi'_s(G) \leq \Delta_A \Delta_B$.

Verified when:

- $\Delta_A = \Delta_B = 3$ [Steger and Yu – 1993]
- $\Delta_A = 2$ [Nakprasit – 2008]

Refined conjecture for bipartite graphs

Conjecture [Brualdi, Quinn Massey – 1993]

If G is (Δ_A, Δ_B)-bipartite, then $\chi'_s(G) \leq \Delta_A \Delta_B$.

Verified when:

- $\Delta_A = \Delta_B = 3$ [Steger and Yu – 1993]
- $\Delta_A = 2$ [Nakprasit – 2008]

We confirm the conjecture when $\Delta_A = 3$ and $\Delta_B \geq 4$

Theorem [B., Lagoutte, Valicov – 2014+]

If G is $(3, \Delta_B)$-bipartite, then $\chi'_s(G) \leq 4\Delta_B$.

$(3, \Delta_B)$-bipartite graphs – a proof

Theorem [B., Lagoutte, Valicov – 2014+]

If G is $(3, \Delta_B)$-bipartite, then $\chi'_s(G) \leq 4\Delta_B$.

Proof. $G = (A, B, E)$, where all vertices in A have degree 3

$(3, \Delta_B)$-bipartite graphs – a proof

Theorem [B., Lagoutte, Valicov – 2014+]

If G is $(3, \Delta_B)$-bipartite, then $\chi'_s(G) \leq 4\Delta_B$.

Proof. $G = (A, B, E)$, where all vertices in A have degree 3

Idea: we produce a strong $4\Delta_B$ edge-coloring c of G by combining an *incidence coloring* c_A of the incidences involving A and one c_B of the incidences involving B

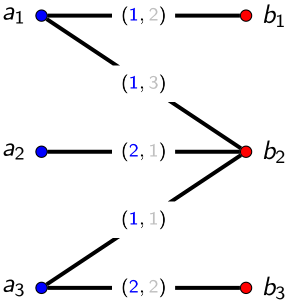
$(3, \Delta_B)$-bipartite graphs – a proof

Theorem [B., Lagoutte, Valicov – 2014+]

If G is $(3, \Delta_B)$-bipartite, then $\chi'_s(G) \leq 4\Delta_B$.

Proof. $G = (A, B, E)$, where all vertices in A have degree 3

Idea: we produce a strong $4\Delta_B$ edge-coloring c of G by combining an *incidence coloring* c_A of the incidences involving A and one c_B of the incidences involving B



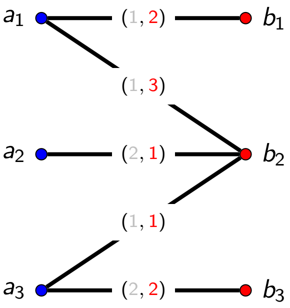
$(3, \Delta_B)$-bipartite graphs – a proof

Theorem [B., Lagoutte, Valicov – 2014+]

If G is $(3, \Delta_B)$-bipartite, then $\chi'_s(G) \leq 4\Delta_B$.

Proof. $G = (A, B, E)$, where all vertices in A have degree 3

Idea: we produce a strong $4\Delta_B$ edge-coloring c of G by combining an *incidence coloring* c_A of the incidences involving A and one c_B of the incidences involving B



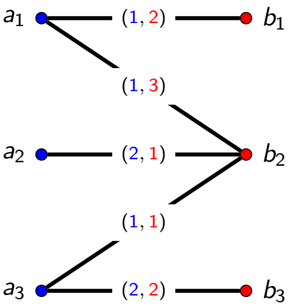
$(3, \Delta_B)$-bipartite graphs – a proof

Theorem [B., Lagoutte, Valicov – 2014+]

If G is $(3, \Delta_B)$-bipartite, then $\chi'_s(G) \leq 4\Delta_B$.

Proof. $G = (A, B, E)$, where all vertices in A have degree 3

Idea: we produce a strong $4\Delta_B$ edge-coloring c of G by combining an *incidence coloring* c_A of the incidences involving A and one c_B of the incidences involving B



Mixing c_A and c_B to get c

Three steps:

Mixing c_A and c_B to get c

Three steps:

1. choose c_B as a *specific* Δ_B-incidence coloring

Mixing c_A and c_B to get c

Three steps:

1. choose c_B as a *specific* Δ_B-incidence coloring
2. according to c_B, define c_A using at most 4 colors

Mixing c_A and c_B to get c

Three steps:

1. choose c_B as a *specific* Δ_B-incidence coloring
2. according to c_B, define c_A using at most 4 colors
3. mix c_A and c_B, *i.e.* set $c(e) = (c_A(e), c_B(e))$ for every $e \in E$

Mixing c_A and c_B to get c

Three steps:

1. choose c_B as a *specific* Δ_B-incidence coloring
2. according to c_B, define c_A using at most 4 colors
3. mix c_A and c_B, *i.e.* set $c(e) = (c_A(e), c_B(e))$ for every $e \in E$

Remark: we have $c(e) \neq c(f)$ as soon as $c_A(e) \neq c_A(f)$ or $c_B(e) \neq c_B(f)$

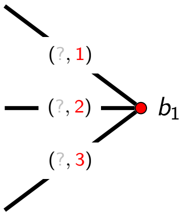
Mixing c_A and c_B to get c

Three steps:

1. choose c_B as a *specific* Δ_B-incidence coloring
2. according to c_B, define c_A using at most 4 colors
3. mix c_A and c_B, *i.e.* set $c(e) = (c_A(e), c_B(e))$ for every $e \in E$

Remark: we have $c(e) \neq c(f)$ as soon as $c_A(e) \neq c_A(f)$ or $c_B(e) \neq c_B(f)$

As c_B, just consider a proper Δ_B-incidence coloring



(adjacent incidences are of the form (b_1, e) and (b_1, f))

Coloring procedure

c_B yields a characterization of the vertices in A as follows:

Coloring procedure

c_B yields a characterization of the vertices in A as follows:

- **Type 1**: all three incident edges have the same color by c_B

Coloring procedure

c_B yields a characterization of the vertices in A as follows:

- **Type 1**: all three incident edges have the same color by c_B
- **Type 2**: two incident edges (= *paired*) have the same color by c_B, which is different from the color of the third one (= *lonely*)

Coloring procedure

c_B yields a characterization of the vertices in A as follows:

- **Type 1**: all three incident edges have the same color by c_B
- **Type 2**: two incident edges (= *paired*) have the same color by c_B, which is different from the color of the third one (= *lonely*)
- **Type 3**: all three incident edges have distinct colors by c_B

Coloring procedure

c_B yields a characterization of the vertices in A as follows:

- **Type 1**: all three incident edges have the same color by c_B
- **Type 2**: two incident edges (= *paired*) have the same color by c_B, which is different from the color of the third one (= *lonely*)
- **Type 3**: all three incident edges have distinct colors by c_B

As c_B, choose the one maximizing the number of Type 1 vertices, and then maximizing the number of Type 2 vertices

Coloring procedure

c_B yields a characterization of the vertices in A as follows:

- **Type 1**: all three incident edges have the same color by c_B
- **Type 2**: two incident edges (= *paired*) have the same color by c_B, which is different from the color of the third one (= *lonely*)
- **Type 3**: all three incident edges have distinct colors by c_B

As c_B, choose the one maximizing the number of Type 1 vertices, and then maximizing the number of Type 2 vertices

For every edge e, we assign a color to $c_A(e)$ in such a way that no conflict appears

Coloring procedure

c_B yields a characterization of the vertices in A as follows:

- **Type 1**: all three incident edges have the same color by c_B
- **Type 2**: two incident edges (= *paired*) have the same color by c_B, which is different from the color of the third one (= *lonely*)
- **Type 3**: all three incident edges have distinct colors by c_B

As c_B, choose the one maximizing the number of Type 1 vertices, and then maximizing the number of Type 2 vertices

For every edge e, we assign a color to $c_A(e)$ in such a way that no conflict appears

Coloring procedure:

Step 1: color the edges incident to Type 1 vertices
Step 2: color the paired edges incident to Type 2
Step 3: color the edges incident to Type 3 vertices
Step 4: color the lonely edges incident to Type 2 vertices

Step 1: color the edges incident to Type 1 vertices

Just assign the colors among $\{1, 2, 3\}$ greedily

Lemma

There is at least one available color for every edge to color.

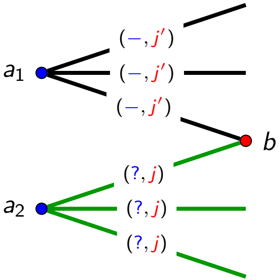
Step 1: color the edges incident to Type 1 vertices

Just assign the colors among $\{1, 2, 3\}$ greedily

Lemma

There is at least one available color for every edge to color.

Proof. Follows from the properness of c_B



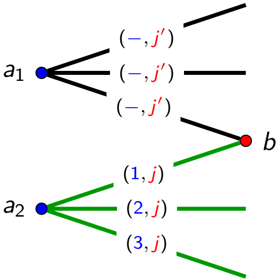
Step 1: color the edges incident to Type 1 vertices

Just assign the colors among $\{1, 2, 3\}$ greedily

Lemma

There is at least one available color for every edge to color.

Proof. Follows from the properness of c_B ∎



Step 2: color the paired edges incident to Type 2 vertices

Just assign the colors among $\{1, 2, 3\}$ greedily

Lemma

There is at least one available color for every edge to color.

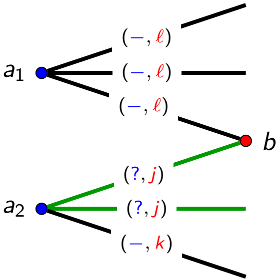
Step 2: color the paired edges incident to Type 2 vertices

Just assign the colors among $\{1, 2, 3\}$ greedily

Lemma

There is at least one available color for every edge to color.

Proof. There may be a forbidden color $(-, j)$ adjacent to the bottom-most edge – this is the only one since c_B is proper and "maximum"



if a_1 is Type 1, then the colors are different

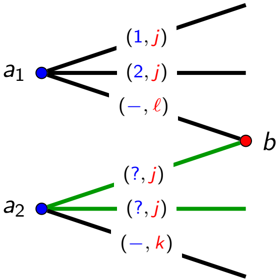
Step 2: color the paired edges incident to Type 2 vertices

Just assign the colors among $\{1, 2, 3\}$ greedily

Lemma

There is at least one available color for every edge to color.

Proof. There may be a forbidden color $(-, j)$ adjacent to the bottom-most edge – this is the only one since c_B is proper and "maximum"



if a_1 is Type 2...

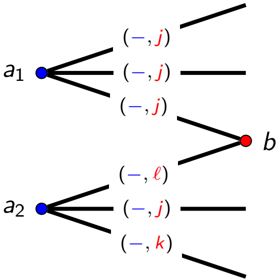
Step 2: color the paired edges incident to Type 2 vertices

Just assign the colors among $\{1, 2, 3\}$ greedily

Lemma

There is at least one available color for every edge to color.

Proof. There may be a forbidden color $(-, j)$ adjacent to the bottom-most edge – this is the only one since c_B is proper and "maximum"



... we could just switch two colors and make a_1 Type 1

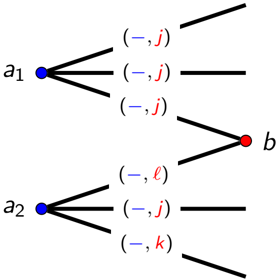
Step 2: color the paired edges incident to Type 2 vertices

Just assign the colors among $\{1, 2, 3\}$ greedily

Lemma

There is at least one available color for every edge to color.

Proof. There may be a forbidden color $(-, j)$ adjacent to the bottom-most edge – this is the only one since c_B is proper and "maximum" ■



... we could just switch two colors and make a_1 Type 1

Step 3: color the edges incident to Type 3 vertices

Just assign the colors among $\{1, 2, 3\}$ greedily

Lemma

There is at least one available color for every edge to color.

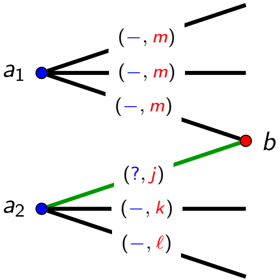
Step 3: color the edges incident to Type 3 vertices

Just assign the colors among $\{1, 2, 3\}$ greedily

Lemma

There is at least one available color for every edge to color.

Proof. There may be up to two forbidden colors $(-, j)$ near the bottom-most edges – these are the only ones since c_B is proper and "maximum"



if a_1 is Type 1, then the colors are different

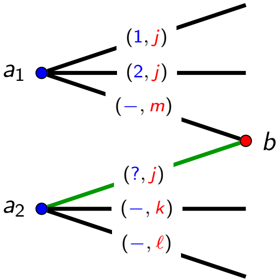
Step 3: color the edges incident to Type 3 vertices

Just assign the colors among $\{1, 2, 3\}$ greedily

Lemma

There is at least one available color for every edge to color.

Proof. There may be up to two forbidden colors $(-, j)$ near the bottom-most edges – these are the only ones since c_B is proper and "maximum"



if a_1 is Type 2...

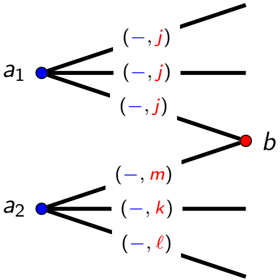
Step 3: color the edges incident to Type 3 vertices

Just assign the colors among $\{1, 2, 3\}$ greedily

Lemma

There is at least one available color for every edge to color.

Proof. There may be up to two forbidden colors $(-, j)$ near the bottom-most edges – these are the only ones since c_B is proper and "maximum"



... we could switch two colors and make a_1 Type 1

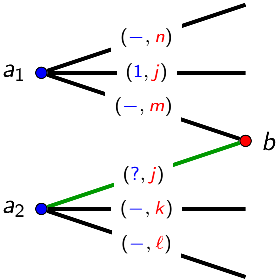
Step 3: color the edges incident to Type 3 vertices

Just assign the colors among $\{1, 2, 3\}$ greedily

Lemma

There is at least one available color for every edge to color.

Proof. There may be up to two forbidden colors $(-, j)$ near the bottom-most edges – these are the only ones since c_B is proper and "maximum"



if a_1 is Type 3...

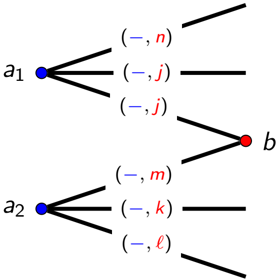
Step 3: color the edges incident to Type 3 vertices

Just assign the colors among $\{1, 2, 3\}$ greedily

Lemma

There is at least one available color for every edge to color.

Proof. There may be up to two forbidden colors $(-, j)$ near the bottom-most edges – these are the only ones since c_B is proper and "maximum"



... we could switch two colors and make a_1 Type 2

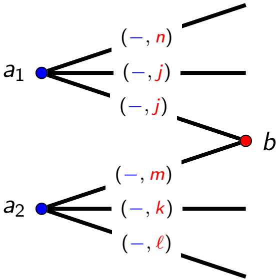
Step 3: color the edges incident to Type 3 vertices

Just assign the colors among $\{1, 2, 3\}$ greedily

Lemma

There is at least one available color for every edge to color.

Proof. There may be up to two forbidden colors $(-, j)$ near the bottom-most edges – these are the only ones since c_B is proper and "maximum" ∎



... we could switch two colors and make a_1 Type 2

Step 4: color the lonely edges incident to Type 2 vertices

$\mathcal{C}_j$: (connected) subgraph induced by the j-lonely edges (*i.e.* with $c_B = j$)

Alternate cycle of $\mathcal{C}_j$: edges alternate between j-lonely and non-j-lonely

Step 4: color the lonely edges incident to Type 2 vertices

$\mathcal{C}_j$: (connected) subgraph induced by the j-lonely edges (*i.e.* with $c_B = j$)

Alternate cycle of $\mathcal{C}_j$: edges alternate between j-lonely and non-j-lonely

Lemma

Every cycle of $\mathcal{C}_j$ is alternate.

Step 4: color the lonely edges incident to Type 2 vertices

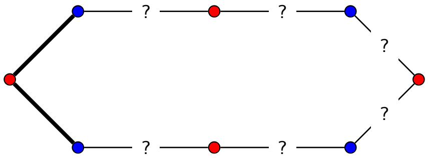
$\mathcal{C}_j$: (connected) subgraph induced by the j-lonely edges (*i.e.* with $c_B = j$)

Alternate cycle of $\mathcal{C}_j$: edges alternate between j-lonely and non-j-lonely

Lemma

Every cycle of $\mathcal{C}_j$ is alternate.

Proof. Assume C is a cycle of $\mathcal{C}_j$ – if C is not alternate, then there are two adjacent non-lonely edges e and e' both incident to a vertex in B



Step 4: color the lonely edges incident to Type 2 vertices

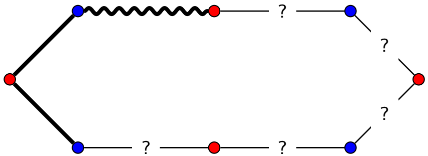
$\mathcal{C}_j$: (connected) subgraph induced by the j-lonely edges (*i.e.* with $c_B = j$)

Alternate cycle of $\mathcal{C}_j$: edges alternate between j-lonely and non-j-lonely

Lemma

Every cycle of $\mathcal{C}_j$ is alternate.

Proof. Assume C is a cycle of $\mathcal{C}_j$ – if C is not alternate, then there are two adjacent non-lonely edges e and e' both incident to a vertex in B



Type 2 + 2 adjacent j-lonely = new Type 1

Step 4: color the lonely edges incident to Type 2 vertices

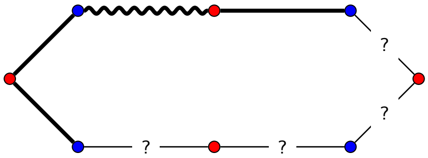
$\mathcal{C}_j$: (connected) subgraph induced by the j-lonely edges (*i.e.* with $c_B = j$)

Alternate cycle of $\mathcal{C}_j$: edges alternate between j-lonely and non-j-lonely

Lemma

Every cycle of $\mathcal{C}_j$ is alternate.

Proof. Assume C is a cycle of $\mathcal{C}_j$ – if C is not alternate, then there are two adjacent non-lonely edges e and e' both incident to a vertex in B



No two adjacent j-lonely

Step 4: color the lonely edges incident to Type 2 vertices

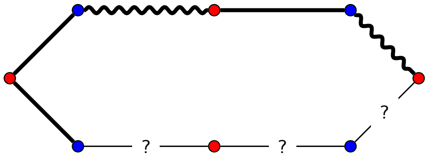
$\mathcal{C}_j$: (connected) subgraph induced by the j-lonely edges (*i.e.* with $c_B = j$)

Alternate cycle of $\mathcal{C}_j$: edges alternate between j-lonely and non-j-lonely

Lemma

Every cycle of $\mathcal{C}_j$ is alternate.

Proof. Assume C is a cycle of $\mathcal{C}_j$ – if C is not alternate, then there are two adjacent non-lonely edges e and e' both incident to a vertex in B



Step 4: color the lonely edges incident to Type 2 vertices

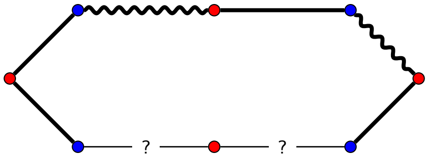
$\mathcal{C}_j$: (connected) subgraph induced by the j-lonely edges (*i.e.* with $c_B = j$)

Alternate cycle of $\mathcal{C}_j$: edges alternate between j-lonely and non-j-lonely

Lemma

Every cycle of $\mathcal{C}_j$ is alternate.

Proof. Assume C is a cycle of $\mathcal{C}_j$ – if C is not alternate, then there are two adjacent non-lonely edges e and e' both incident to a vertex in B



Step 4: color the lonely edges incident to Type 2 vertices

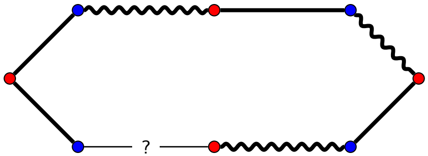
$\mathcal{C}_j$: (connected) subgraph induced by the j-lonely edges (*i.e.* with $c_B = j$)

Alternate cycle of $\mathcal{C}_j$: edges alternate between j-lonely and non-j-lonely

Lemma

Every cycle of $\mathcal{C}_j$ is alternate.

Proof. Assume C is a cycle of $\mathcal{C}_j$ – if C is not alternate, then there are two adjacent non-lonely edges e and e' both incident to a vertex in B



Step 4: color the lonely edges incident to Type 2 vertices

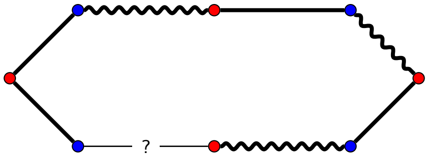
$\mathcal{C}_j$: (connected) subgraph induced by the j-lonely edges (*i.e.* with $c_B = j$)

Alternate cycle of $\mathcal{C}_j$: edges alternate between j-lonely and non-j-lonely

Lemma

Every cycle of $\mathcal{C}_j$ is alternate.

Proof. Assume C is a cycle of $\mathcal{C}_j$ – if C is not alternate, then there are two adjacent non-lonely edges e and e' both incident to a vertex in B



How to close the tour?

Step 4: color the lonely edges incident to Type 2 vertices

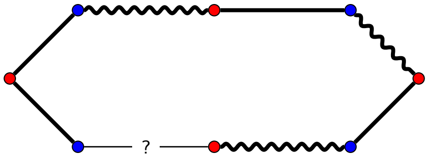
$\mathcal{C}_j$: (connected) subgraph induced by the j-lonely edges (*i.e.* with $c_B = j$)

Alternate cycle of $\mathcal{C}_j$: edges alternate between j-lonely and non-j-lonely

Lemma

Every cycle of $\mathcal{C}_j$ is alternate.

Proof. Assume C is a cycle of $\mathcal{C}_j$ – if C is not alternate, then there are two adjacent non-lonely edges e and e' both incident to a vertex in B ■



How to close the tour?

On the structure of the $\mathcal{C}_j$'s

Lemma

Every two cycles of $\mathcal{C}_j$ are disjoint.

On the structure of the $\mathcal{C}_j$'s

Lemma

Every two cycles of $\mathcal{C}_j$ are disjoint.

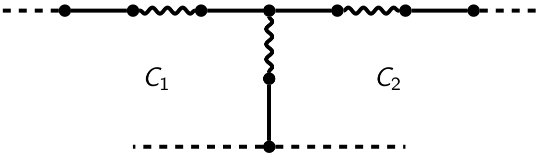
Proof. If two cycles C_1 and C_2 of $\mathcal{C}_j$ share a vertex without sharing an edge, then a vertex is adjacent to two j-lonely edges

On the structure of the $\mathcal{C}_j$'s

Lemma

Every two cycles of $\mathcal{C}_j$ are disjoint.

Proof. If two cycles C_1 and C_2 of $\mathcal{C}_j$ share a vertex without sharing an edge, then a vertex is adjacent to two j-lonely edges

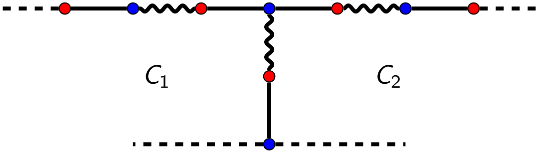


On the structure of the $\mathcal{C}_j$'s

Lemma

Every two cycles of $\mathcal{C}_j$ are disjoint.

Proof. If two cycles C_1 and C_2 of $\mathcal{C}_j$ share a vertex without sharing an edge, then a vertex is adjacent to two j-lonely edges



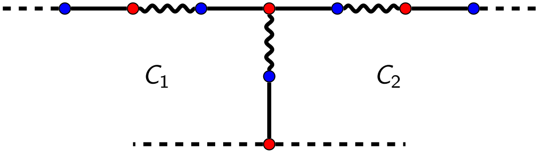
Type 2 + 2 adjacent lonely = new Type 1

On the structure of the $\mathcal{C}_j$'s

Lemma

Every two cycles of $\mathcal{C}_j$ are disjoint.

Proof. If two cycles C_1 and C_2 of $\mathcal{C}_j$ share a vertex without sharing an edge, then a vertex is adjacent to two j-lonely edges



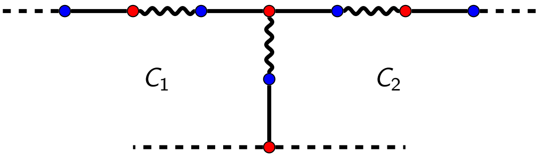
Second end point of the intersecting path cannot be correct

On the structure of the $\mathcal{C}_j$'s

Lemma

Every two cycles of $\mathcal{C}_j$ are disjoint.

Proof. If two cycles C_1 and C_2 of $\mathcal{C}_j$ share a vertex without sharing an edge, then a vertex is adjacent to two j-lonely edges ■



Second end point of the intersecting path cannot be correct

On the structure of the $\mathcal{C}_j$'s

Lemma

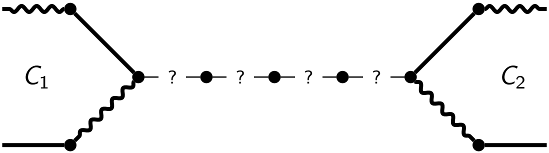
$\mathcal{C}_j$ cannot have two disjoint cycles joined by a path.

On the structure of the $\mathcal{C}_j$'s

Lemma

$\mathcal{C}_j$ cannot have two disjoint cycles joined by a path.

Proof. Assume C_1 and C_2 are linked by a path $u ... v$ where $u \in C_1$ and $v \in C_2$

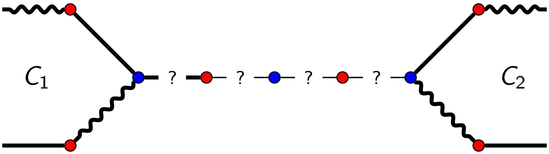


On the structure of the $\mathcal{C}_j$'s

Lemma

$\mathcal{C}_j$ cannot have two disjoint cycles joined by a path.

Proof. Assume C_1 and C_2 are linked by a path $u \ldots v$ where $u \in C_1$ and $v \in C_2$



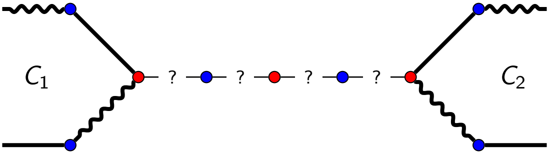
Type 2 + 2 adjacent lonely = new Type 1

On the structure of the $\mathcal{C}_j$'s

Lemma

$\mathcal{C}_j$ cannot have two disjoint cycles joined by a path.

Proof. Assume C_1 and C_2 are linked by a path $u ... v$ where $u \in C_1$ and $v \in C_2$

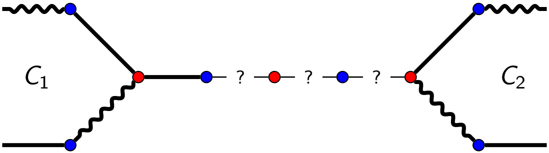


On the structure of the $\mathcal{C}_j$'s

Lemma

$\mathcal{C}_j$ cannot have two disjoint cycles joined by a path.

Proof. Assume C_1 and C_2 are linked by a path $u \ldots v$ where $u \in C_1$ and $v \in C_2$

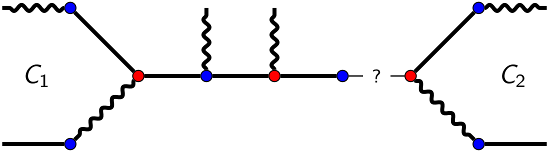


On the structure of the $\mathcal{C}_j$'s

Lemma

$\mathcal{C}_j$ cannot have two disjoint cycles joined by a path.

Proof. Assume C_1 and C_2 are linked by a path $u \ldots v$ where $u \in C_1$ and $v \in C_2$



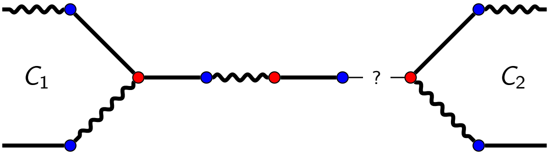
Type 2 + 2 adjacent lonely = new Type 1

On the structure of the $\mathcal{C}_j$'s

Lemma

$\mathcal{C}_j$ cannot have two disjoint cycles joined by a path.

Proof. Assume C_1 and C_2 are linked by a path $u...v$ where $u \in C_1$ and $v \in C_2$



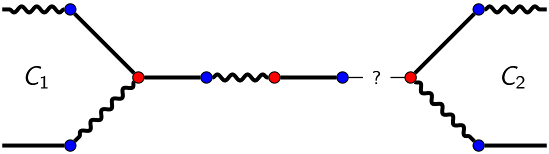
How to join the two cycles?

On the structure of the $\mathcal{C}_j$'s

Lemma

$\mathcal{C}_j$ cannot have two disjoint cycles joined by a path.

Proof. Assume C_1 and C_2 are linked by a path $u \ldots v$ where $u \in C_1$ and $v \in C_2$ ■



How to join the two cycles?

Back to coloring Step 4

Step 4:

Phase 1: color the unique cycle C of $\mathcal{C}_j$

Phase 2: color every tree T of the forest $\mathcal{C}_j - E(C)$

Back to coloring Step 4

Step 4:

Phase 1: color the unique cycle C of $\mathcal{C}_j$
Phase 2: color every tree T of the forest $\mathcal{C}_j - E(C)$

Lemma

The j-lonely edges of C can be colored with $\{1, 2, 3, 4\}$.

Back to coloring Step 4

Step 4:

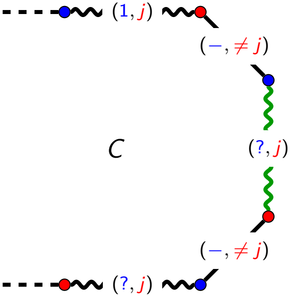
Phase 1: color the unique cycle C of $\mathcal{C}_j$

Phase 2: color every tree T of the forest $\mathcal{C}_j - E(C)$

Lemma

The j-lonely edges of C can be colored with $\{1, 2, 3, 4\}$.

Proof. Color the j-lonely edges consecutively



Back to coloring Step 4

Step 4:

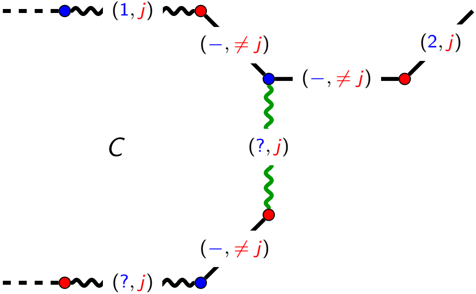
Phase 1: color the unique cycle C of $\mathcal{C}_j$

Phase 2: color every tree T of the forest $\mathcal{C}_j - E(C)$

Lemma

The j-lonely edges of C can be colored with $\{1, 2, 3, 4\}$.

Proof. Color the j-lonely edges consecutively



Back to coloring Step 4

Step 4:

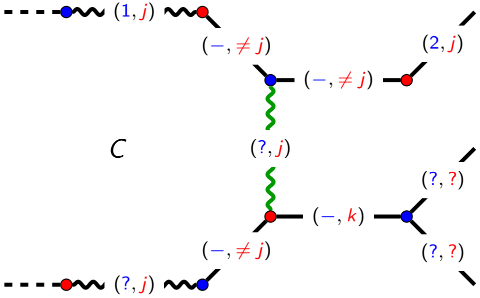
Phase 1: color the unique cycle C of $\mathcal{C}_j$

Phase 2: color every tree T of the forest $\mathcal{C}_j - E(C)$

Lemma

The j-lonely edges of C can be colored with $\{1, 2, 3, 4\}$.

Proof. Color the j-lonely edges consecutively



Back to coloring Step 4

Step 4:

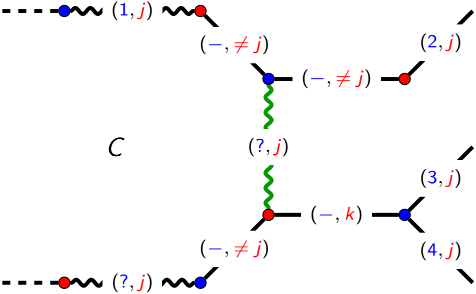
Phase 1: color the unique cycle C of $\mathcal{C}_j$

Phase 2: color every tree T of the forest $\mathcal{C}_j - E(C)$

Lemma

The j-lonely edges of C can be colored with $\{1, 2, 3, 4\}$.

Proof. Color the j-lonely edges consecutively



Switch to create a new Type 1 vertex

Back to coloring Step 4

Step 4:

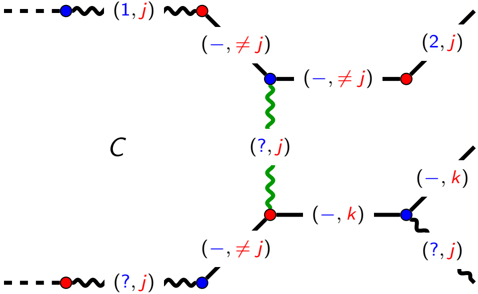
Phase 1: color the unique cycle C of $\mathcal{C}_j$

Phase 2: color every tree T of the forest $\mathcal{C}_j - E(C)$

Lemma

The j-lonely edges of C can be colored with $\{1, 2, 3, 4\}$.

Proof. Color the j-lonely edges consecutively



Not yet colored

Back to coloring Step 4

Step 4:

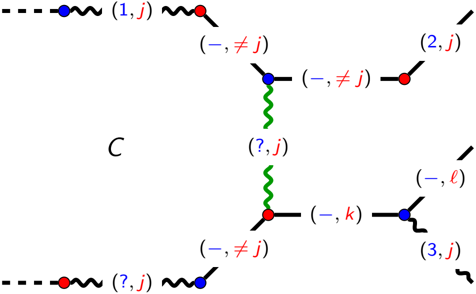
Phase 1: color the unique cycle C of $\mathcal{C}_j$

Phase 2: color every tree T of the forest $\mathcal{C}_j - E(C)$

Lemma

The j-lonely edges of C can be colored with $\{1, 2, 3, 4\}$.

Proof. Color the j-lonely edges consecutively



Switch to create a new Type 2 vertex

Back to coloring Step 4

Step 4:

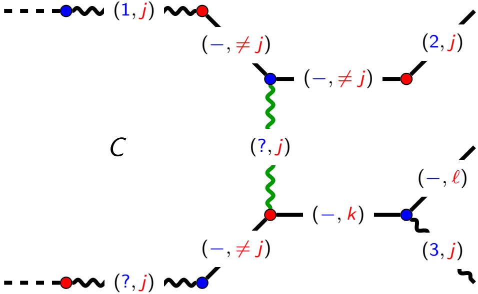
Phase 1: color the unique cycle C of $\mathcal{C}_j$

Phase 2: color every tree T of the forest $\mathcal{C}_j - E(C)$

Lemma

The j-lonely edges of C can be colored with $\{1, 2, 3, 4\}$.

Proof. Color the j-lonely edges consecutively ■



Switch to create a new Type 2 vertex

Step 4 – the end!

Remark: color 4 may be needed for the last edge of C

Step 4 – the end!

Remark: color 4 may be needed for the last edge of C

Lemma

The j-lonely edges of T can be colored with $\{1, 2, 3, 4\}$.

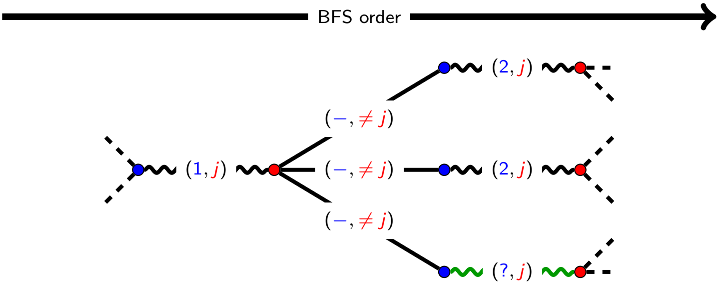
Step 4 – the end!

Remark: color 4 may be needed for the last edge of C

Lemma

The j-lonely edges of T can be colored with $\{1, 2, 3, 4\}$.

Proof. Color the j-lonely edges as given by a BFS algorithm



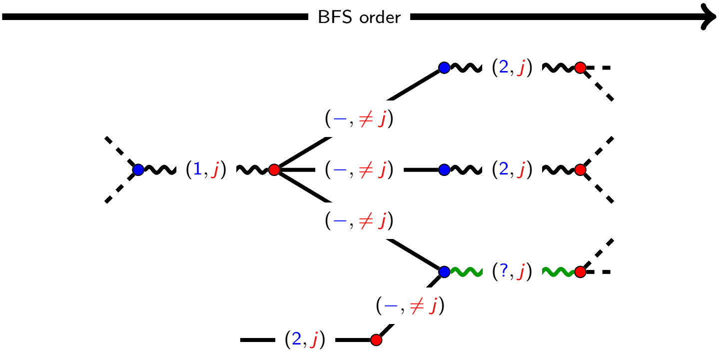
Step 4 – the end!

Remark: color 4 may be needed for the last edge of C

Lemma

The j-lonely edges of T can be colored with $\{1, 2, 3, 4\}$.

Proof. Color the j-lonely edges as given by a BFS algorithm



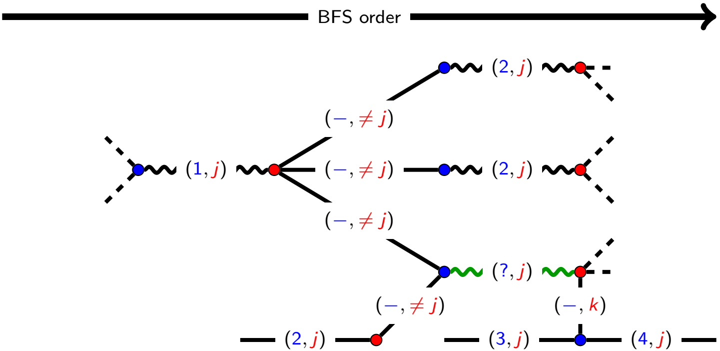
Step 4 – the end!

Remark: color 4 may be needed for the last edge of C

Lemma

The j-lonely edges of T can be colored with $\{1, 2, 3, 4\}$.

Proof. Color the j-lonely edges as given by a BFS algorithm



Switch to create a new Type 1 vertex

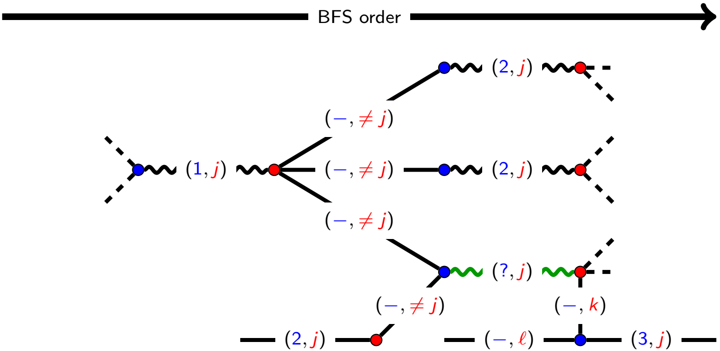
Step 4 – the end!

Remark: color 4 may be needed for the last edge of C

Lemma

The j-lonely edges of T can be colored with $\{1, 2, 3, 4\}$.

Proof. Color the j-lonely edges as given by a BFS algorithm



Switch to create a new Type 2 vertex

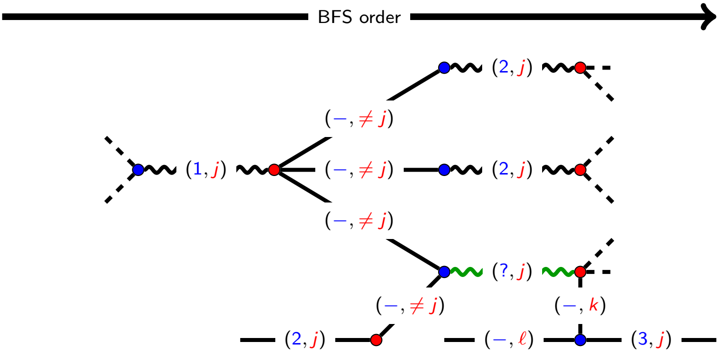
Step 4 – the end!

Remark: color 4 may be needed for the last edge of C

Lemma

The j-lonely edges of T can be colored with $\{1, 2, 3, 4\}$.

Proof. Color the j-lonely edges as given by a BFS algorithm ∎



Switch to create a new Type 2 vertex

Conclusions and open questions

- The refined conjecture says $3\Delta_B$...
- ... can our proof be improved?

Conclusions and open questions

- The refined conjecture says $3\Delta_B$...
- ... can our proof be improved?
- Hardly generalizable to larger values of Δ_A...
- ... though it might be successful for 4

Conclusions and open questions

- The refined conjecture says $3\Delta_B$...
- ... can our proof be improved?
- Hardly generalizable to larger values of Δ_A...
- ... though it might be successful for 4
- Particular construction of c...
- ... what for the list version?

Conclusions and open questions

- The refined conjecture says $3\Delta_B$...
- ... can our proof be improved?
- Hardly generalizable to larger values of Δ_A...
- ... though it might be successful for 4
- Particular construction of c...
- ... what for the list version?
- Everything is done in polynomial time with c_B in hand...
- ... but it is NP-complete to choose it conveniently

Conclusions and open questions

- The refined conjecture says $3\Delta_B$...
- ... can our proof be improved?
- Hardly generalizable to larger values of Δ_A...
- ... though it might be successful for 4
- Particular construction of c...
- ... what for the list version?
- Everything is done in polynomial time with c_B in hand...
- ... but it is NP-complete to choose it conveniently

Thank you for your attention.