Parameterized Complexity of Conflict-free Graph Coloring

Hans L. Bodlaender[1], Sudeshna Kolay[2], and Astrid Pieterse[3]

[1] Utrecht University, The Netherlands
[2] Ben Gurion University of Negev, Israel
[3] Eindhoven University of Technology, The Netherlands

Conflict-free graph coloring

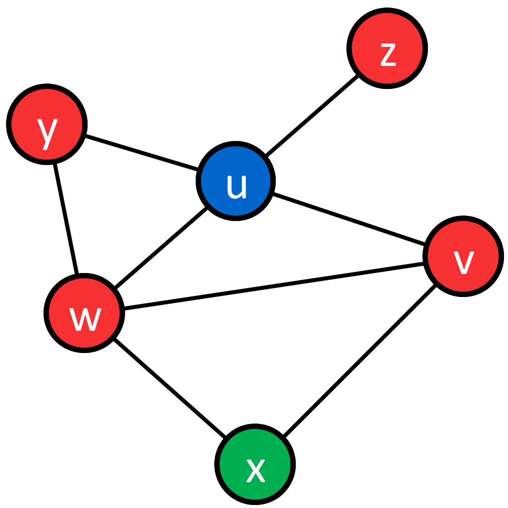
q-Closed Neighborhood Conflict-Free Coloring

q-CNCF-Coloring

Input: A graph G (with vertex cover S)

Parameter: $k = |S|$

Question: Is it possible to assign every vertex in G a color from $\{1, \ldots, q\}$, such that for all v, there is a color occurring exactly once in $N[v]$?



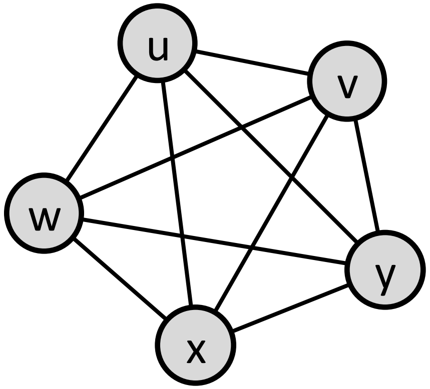
Related: q-ONCF-Coloring

- Considers open neighborhoods instead

CNCF-Coloring

How many colors do we need to CNCF-color

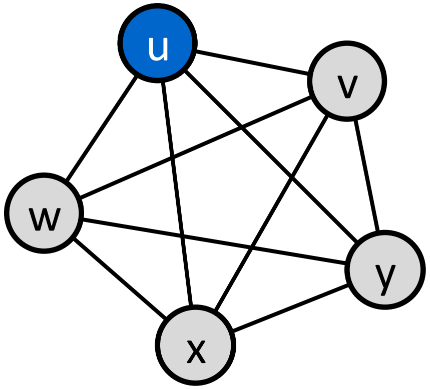
- A clique on k vertices?



CNCF-Coloring

How many colors do we need to CNCF-color

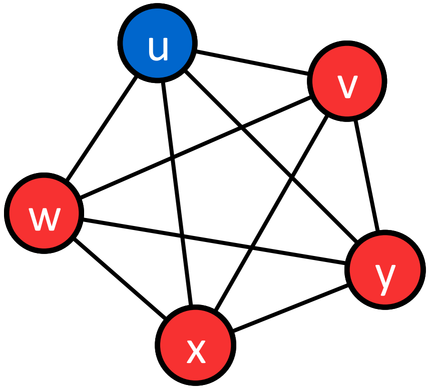
- A clique on k vertices?



CNCF-Coloring

How many colors do we need to CNCF-color

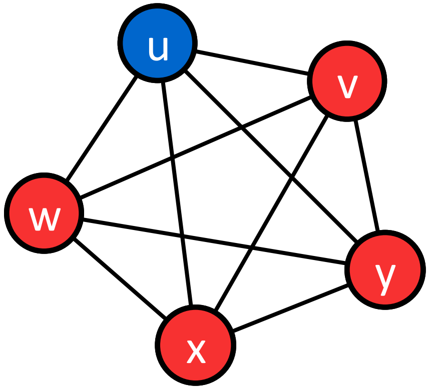
- A clique on k vertices?



CNCF-Coloring

How many colors do we need to CNCF-color

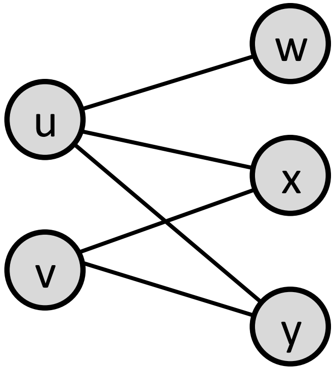
- A clique on k vertices? 2



CNCF-Coloring

How many colors do we need to CNCF-color

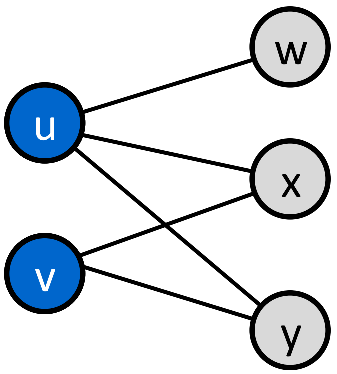
- A clique on k vertices? 2
- A bipartite graph?



CNCF-Coloring

How many colors do we need to CNCF-color

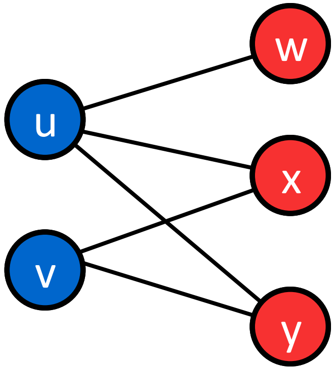
- A clique on k vertices? 2
- A bipartite graph?



CNCF-Coloring

How many colors do we need to CNCF-color

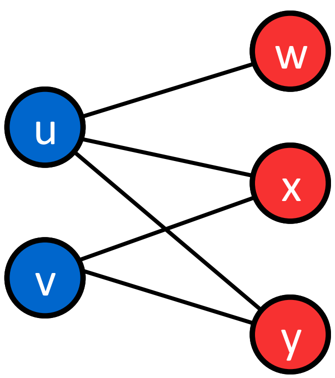
- A clique on k vertices? 2
- A bipartite graph?



CNCF-Coloring

How many colors do we need to CNCF-color

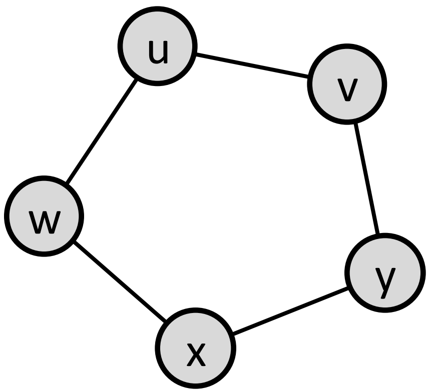
- A clique on k vertices? 2
- A bipartite graph? 2



CNCF-Coloring

How many colors do we need to CNCF-color

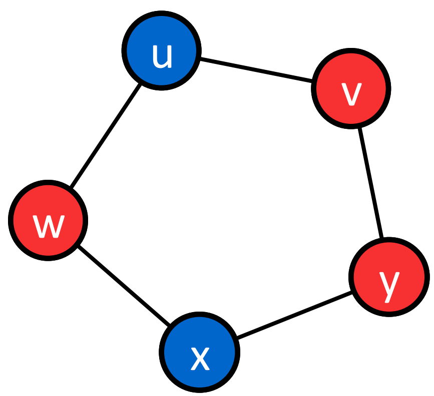
- A clique on k vertices? 2
- A bipartite graph? 2
- An odd cycle?



CNCF-Coloring

How many colors do we need to CNCF-color

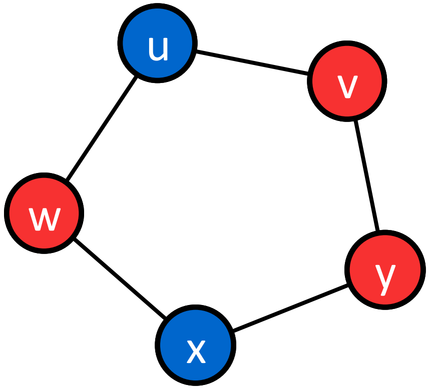
- A clique on k vertices? 2
- A bipartite graph? 2
- An odd cycle?



CNCF-Coloring

How many colors do we need to CNCF-color

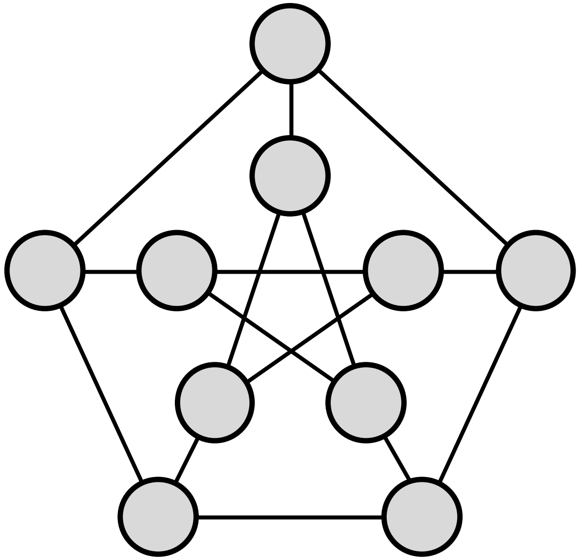
- A clique on k vertices? 2
- A bipartite graph? 2
- An odd cycle? 2



CNCF-Coloring

How many colors do we need to CNCF-color

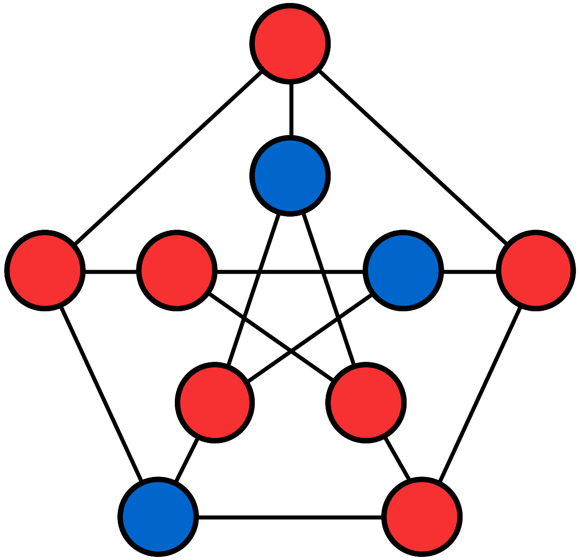
- A clique on k vertices? 2
- A bipartite graph? 2
- An odd cycle? 2
- Petersen graph



CNCF-Coloring

How many colors do we need to CNCF-color

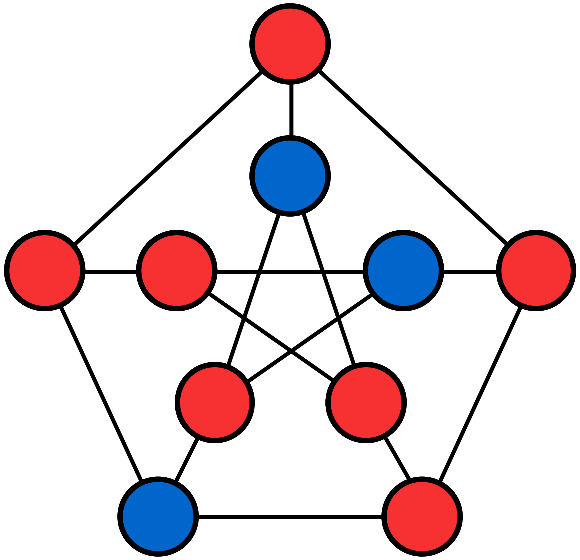
- A clique on k vertices? 2
- A bipartite graph? 2
- An odd cycle? 2
- Petersen graph



CNCF-Coloring

How many colors do we need to CNCF-color

- A clique on k vertices? 2
- A bipartite graph? 2
- An odd cycle? 2
- Petersen graph 2



CNCF-Coloring

How many colors do we need to CNCF-color

- A clique on k vertices? 2
- A bipartite graph? 2
- An odd cycle? 2
- Petersen graph 2
- A k-colorable graph?

CNCF-Coloring

How many colors do we need to CNCF-color

• A clique on k vertices?	2
• A bipartite graph?	2
• An odd cycle?	2
• Petersen graph	2
• A k-colorable graph?	k

Observation

Any proper coloring is a CNCF-Coloring

- Each vertex is the uniquely colored vertex in its closed neighborhood

CNCF-Coloring

How many colors do we need to CNCF-color

• A clique on k vertices?	2	≤ k
• A bipartite graph?	2	≤ 2
• An odd cycle?	2	≤ 3
• Petersen graph	2	≤ 4
• A k-colorable graph?	k	≤ k

Observation

Any proper coloring is a CNCF-Coloring

- Each vertex is the uniquely colored vertex in its closed neighborhood

Background

- Special case of conflict-free coloring set systems
  - Sets given by closed (or open) neighborhoods
- Frequency assignment problems

Studied from a combinatorial perspective

- CNCF-coloring n-vertex graph takes $O(\log^2 n)$ colors [Pach, Tardos 2009]
  - Tight [Glebov, Szabó, Tardos, 2014]

Background

NP-hard for $q \geq 2$ [Gargano and Rescigno, TCS, 2015]

- Study parameterized complexity and kernelizability
  - FPT parameterized by [Gargano and Rescigno, TCS, 2015]
    - Vertex Cover
    - Neighborhood diversity
    - Treewidth
  - In this talk: kernels!

Results

Kernelization parameterized by Vertex Cover

- 2-CNCF-Coloring has a polynomial kernel (up next)

- q-CNCF-Coloring has no polynomial kernel[1] for $q \geq 3$
- q-ONCF-Coloring has no polynomial kernel[1] for $q \geq 2$
  - Both results proven by cross-composition
  - See https://arxiv.org/pdf/1905.00305

[1]Unless $NP \subseteq coNP/\mathrm{poly}$

A general reduction rule

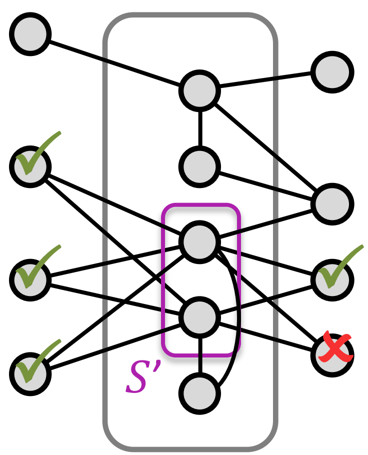
For q-CNCF-Coloring

[Based on Gargano and Rescigno, TCS 2015, Lemma 6]

Reducing number of twins

Let $S' \subseteq S$. Suppose there are $> q+1$ vertices $v \notin S$ with $N(v) = S'$. Mark $q+1$, remove the others.

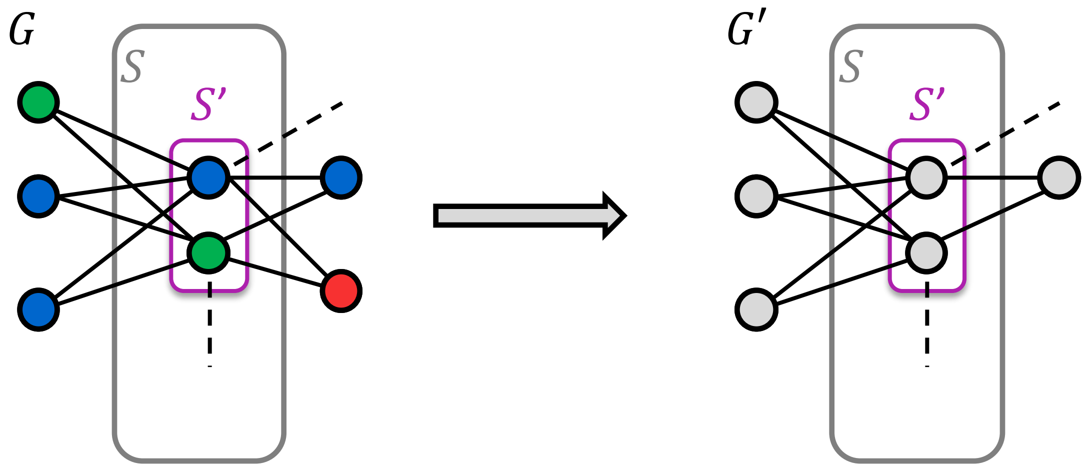
$q = 3$



Correctness

Let G' be the resulting graph

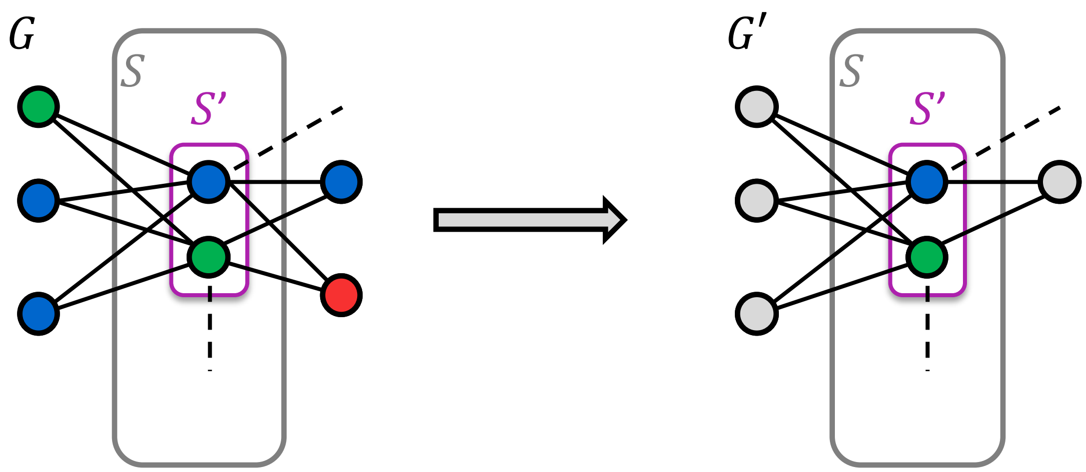
- Suppose G is q-CNCF-Colorable
  - Color G' similarly, ensuring that each vertex in S keeps its conflict-free neighbor (if any)



Correctness

Let G' be the resulting graph

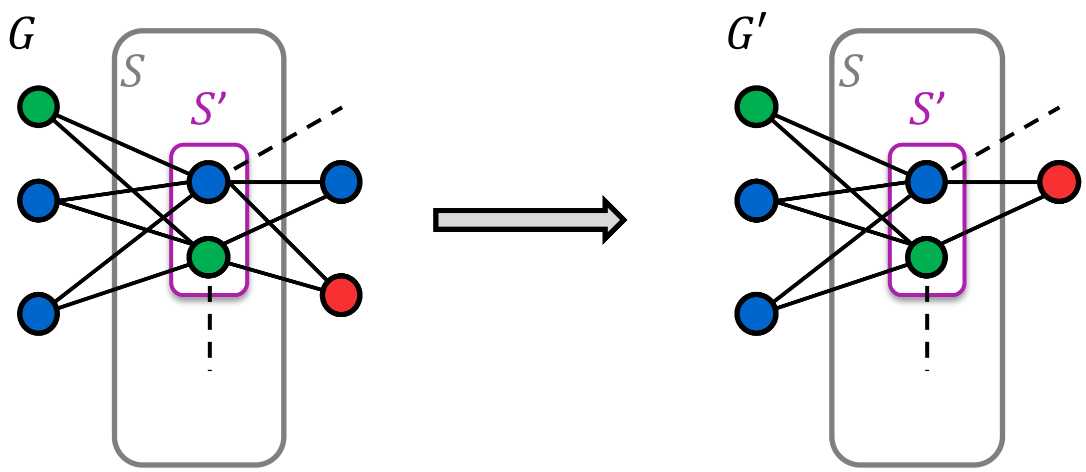
- Suppose G is q-CNCF-Colorable
  - Color G' similarly, ensuring that each vertex in S keeps its conflict-free neighbor (if any)



Correctness

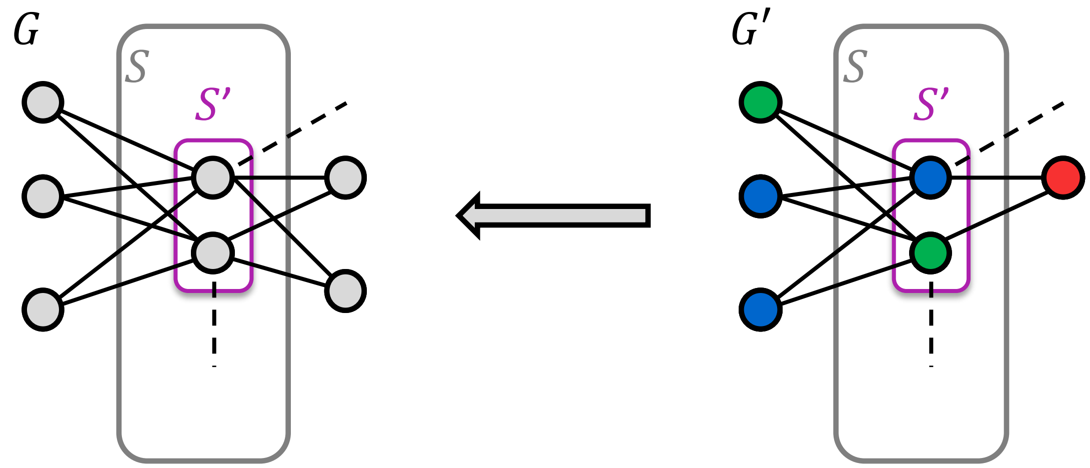
Let G' be the resulting graph

- Suppose G is q-CNCF-Colorable
  - Color G' similarly, ensuring that each vertex in S keeps its conflict-free neighbor (if any)



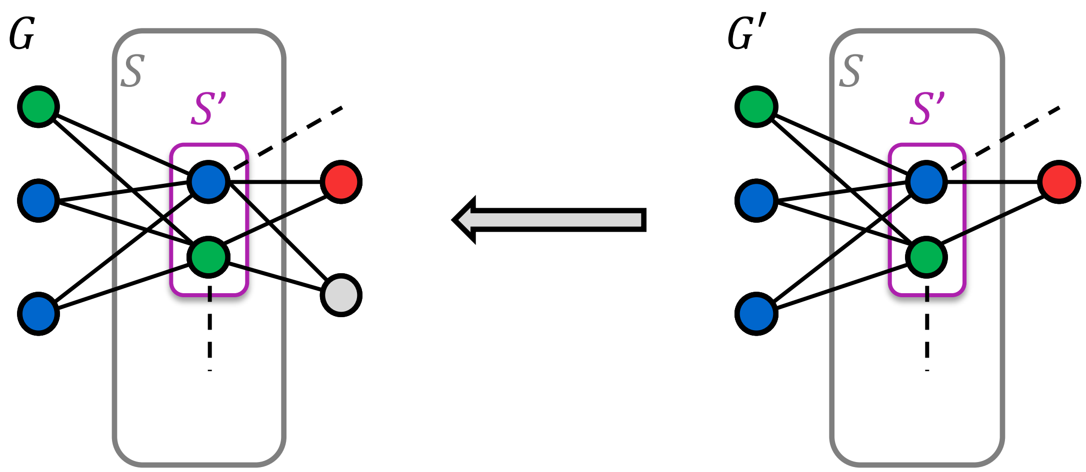
Correctness

- Suppose G' is q-CNCF-Colorable
  - Use the same coloring on G
  - Let v be a removed vertex, color v using a color already used twice in $N(v)$
    - Observe that such a color exists!



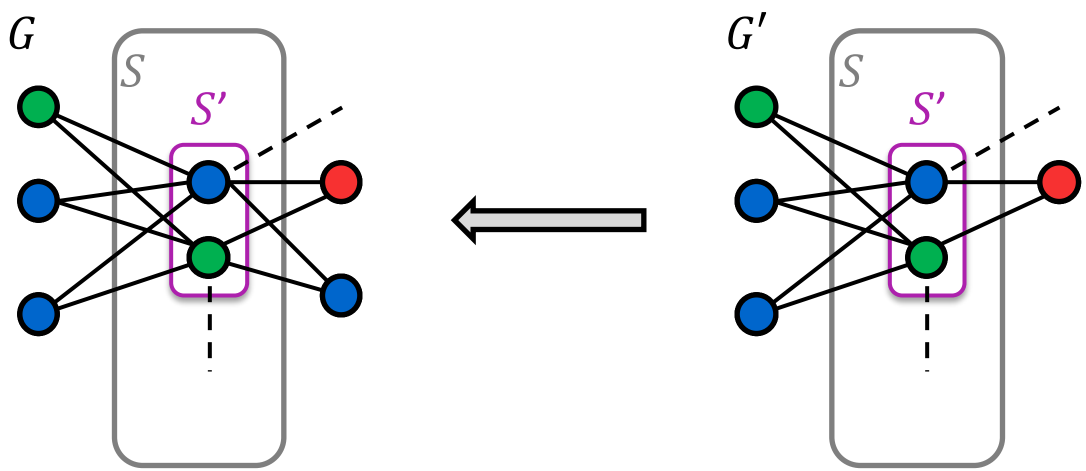
Correctness

- Suppose G' is q-CNCF-Colorable
  - Use the same coloring on G
  - Let v be a removed vertex, color v using a color already used twice in $N(v)$
    - Observe that such a color exists!



Correctness

- Suppose G' is q-CNCF-Colorable
  - Use the same coloring on G
  - Let v be a removed vertex, color v using a color already used twice in $N(v)$
    - Observe that such a color exists!



Effectiveness

Suppose we apply this rule exhaustively, for all $S' \subseteq S$

Number of vertices in S

- k by definition

Number of vertices not in S

- Number of degree-d vertices not in S
  - At most $(q+1)\binom{|S|}{d} = O(k^d)$
- Total $O(2^k)$

Exponential, unless we can somehow bound the number of high-degree vertices

Polynomial kernel for 2-CNCF-Coloring Extension

Parameterized by Vertex Cover size

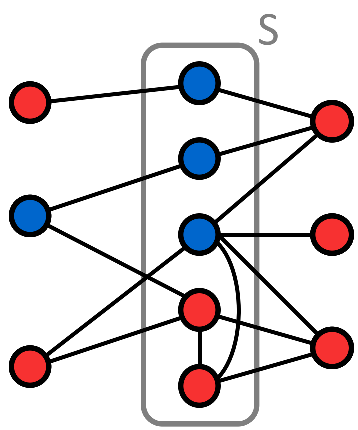
The extension problem

2-CNCF-Coloring Extension

Input: A graph G, with vertex cover S and partial coloring $c : S \to \{red, blue\}$

Parameter: $|S|$

Question: Can c be extended to a 2-CNCF-coloring of G?

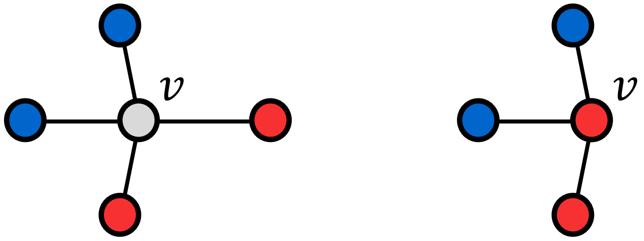


This is a yes-instance!

Trivial reduction rule

If at any point there is a vertex v whose neighborhood contains at least two red and two blue vertices under c

- Output **NO**



Removing low-degree vertices

Apply known reduction rule:

For every $S' \subseteq S$ of size at most 2

- Mark 3 vertices $u \notin S$ such that $N(u) = S'$
  - If there are less than three, mark all

Delete all unmarked vertices of degree 1 and 2

Number of vertices in S

- k (by definition)

Number of degree-≤2 vertices not in S

- At most $3\mathrm{k} + 3\binom{k}{2} = O(k^2)$

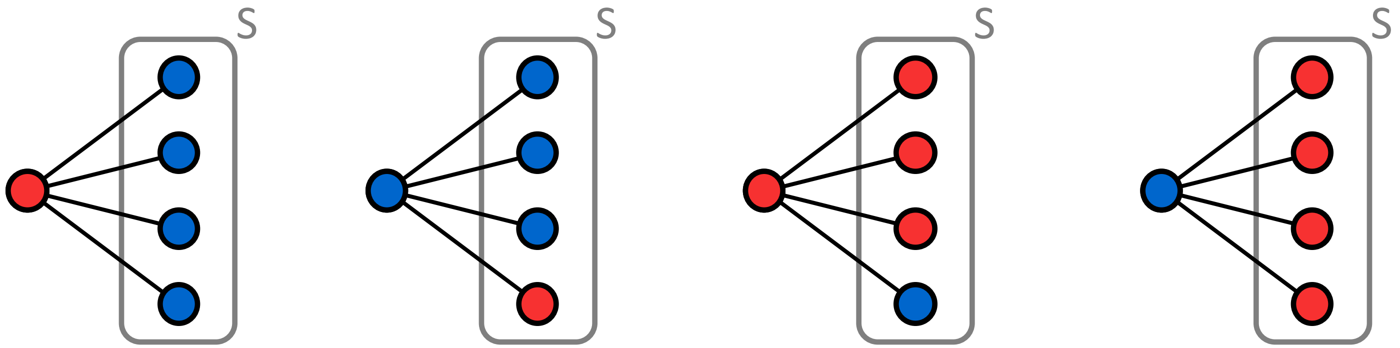
Number of degree-≥3 vertices not in S

- Possibly many ☹

Removing high-degree vertices

For all $v \notin S$ of degree at least 3 do the following.

- Extend c by coloring v such that $N[v]$ is conflict-free
  - This uniquely determines $c(v)$



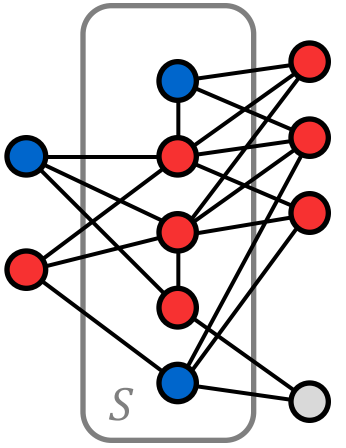
Marking procedure

For every $x \in S$, mark 2 red and 2 blue neighbors in $V(G) \setminus S$

- If there are no two red/blue neighbors, mark all red/blue neighbors

Delete all unmarked vertices

of degree ≥ 3

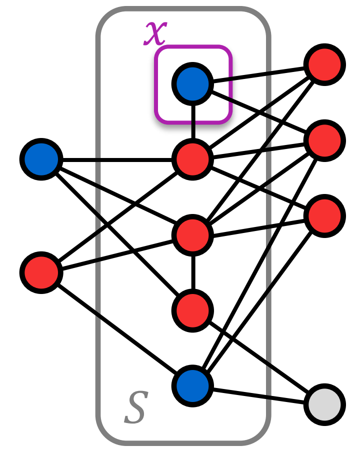


Marking procedure

For every $x \in S$, mark 2 red and 2 blue neighbors in $V(G) \setminus S$

- If there are no two red/blue neighbors, mark all red/blue neighbors

Delete all unmarked vertices of degree ≥ 3

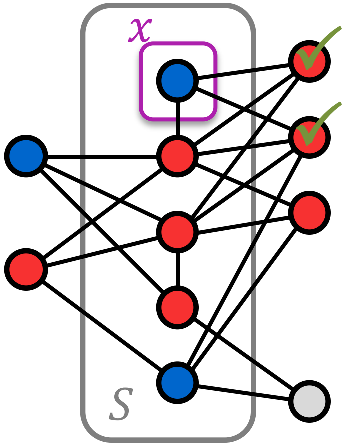


Marking procedure

For every $x \in S$, mark 2 red and 2 blue neighbors in $V(G) \setminus S$

- If there are no two red/blue neighbors, mark all red/blue neighbors

Delete all unmarked vertices of degree ≥ 3

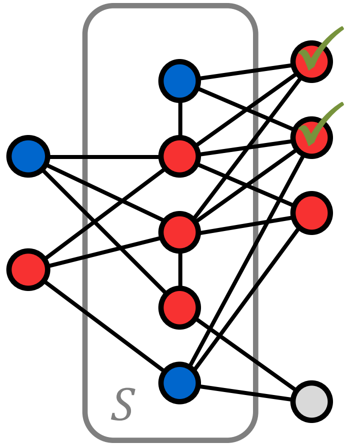


Marking procedure

For every $x \in S$, mark 2 red and 2 blue neighbors in $V(G) \setminus S$

- If there are no two red/blue neighbors, mark all red/blue neighbors

Delete all unmarked vertices of degree ≥ 3

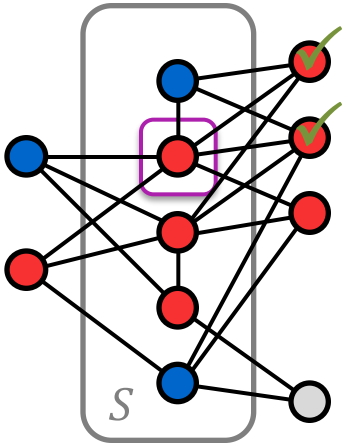


Marking procedure

For every $x \in S$, mark 2 red and 2 blue neighbors in $V(G) \setminus S$

- If there are no two red/blue neighbors, mark all red/blue neighbors

Delete all unmarked vertices of degree ≥ 3

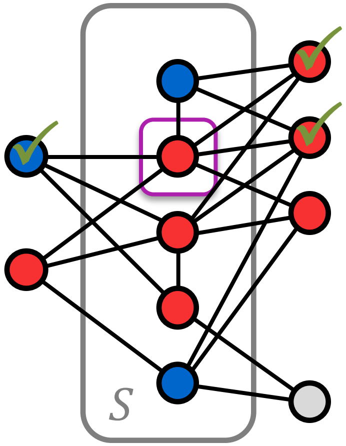


Marking procedure

For every $x \in S$, mark 2 red and 2 blue neighbors in $V(G) \setminus S$

- If there are no two red/blue neighbors, mark all red/blue neighbors

Delete all unmarked vertices of degree ≥ 3



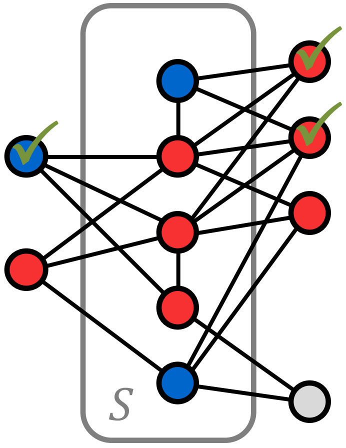
Marking procedure

For every $x \in S$, mark 2 red and 2 blue neighbors in $V(G) \setminus S$

- If there are no two red/blue neighbors, mark all red/blue neighbors

Delete all unmarked vertices

of degree ≥ 3



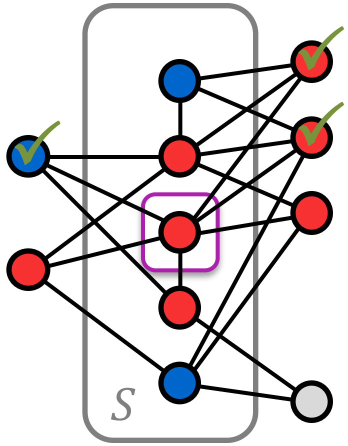
Marking procedure

For every $x \in S$, mark 2 red and 2 blue neighbors in $V(G) \setminus S$

- If there are no two red/blue neighbors, mark all red/blue neighbors

Delete all unmarked vertices

of degree ≥ 3

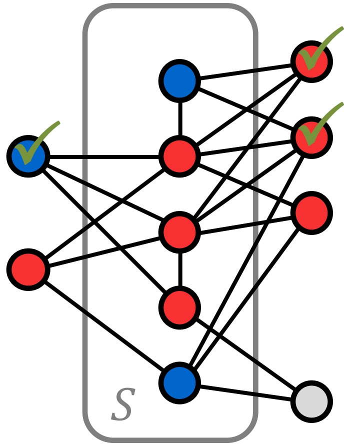


Marking procedure

For every $x \in S$, mark 2 red and 2 blue neighbors in $V(G) \setminus S$

- If there are no two red/blue neighbors, mark all red/blue neighbors

Delete all unmarked vertices of degree ≥ 3

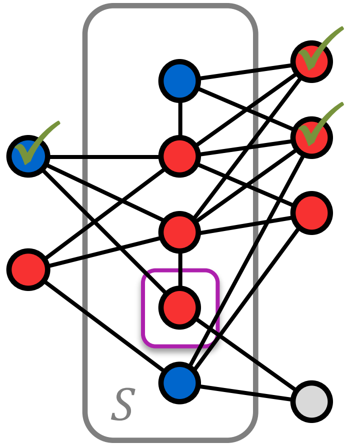


Marking procedure

For every $x \in S$, mark 2 red and 2 blue neighbors in $V(G) \setminus S$

- If there are no two red/blue neighbors, mark all red/blue neighbors

Delete all unmarked vertices of degree ≥ 3



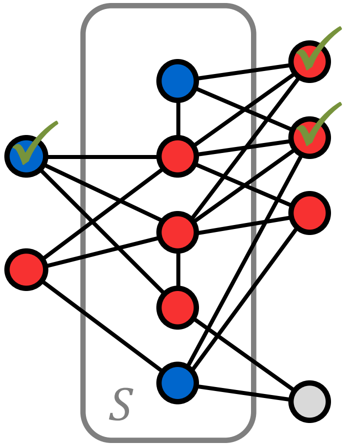
Marking procedure

For every $x \in S$, mark 2 red and 2 blue neighbors in $V(G) \setminus S$

- If there are no two red/blue neighbors, mark all red/blue neighbors

Delete all unmarked vertices

of degree ≥ 3



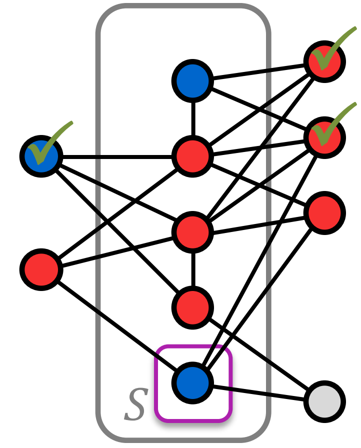
Marking procedure

For every $x \in S$, mark 2 red and 2 blue neighbors in $V(G) \setminus S$

- If there are no two red/blue neighbors, mark all red/blue neighbors

Delete all unmarked vertices

of degree ≥ 3

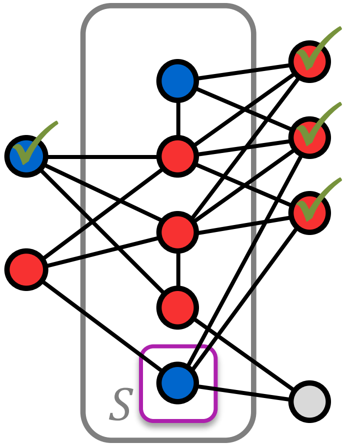


Marking procedure

For every $x \in S$, mark 2 red and 2 blue neighbors in $V(G) \setminus S$

- If there are no two red/blue neighbors, mark all red/blue neighbors

Delete all unmarked vertices of degree ≥ 3

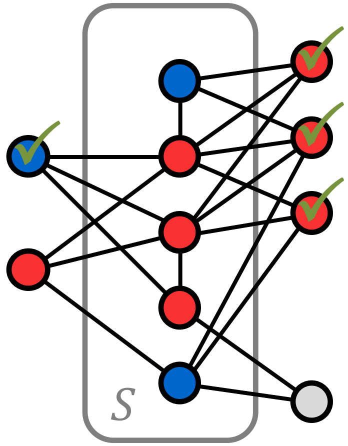


Marking procedure

For every $x \in S$, mark 2 red and 2 blue neighbors in $V(G) \setminus S$

- If there are no two red/blue neighbors, mark all red/blue neighbors

Delete all unmarked vertices of degree ≥ 3



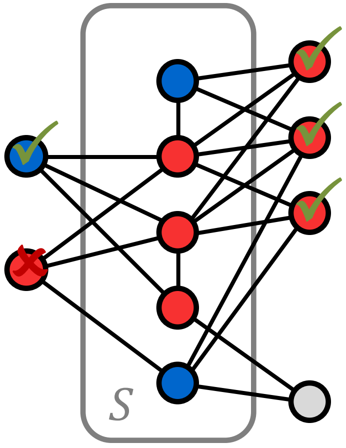
Marking procedure

For every $x \in S$, mark 2 red and 2 blue neighbors in $V(G) \setminus S$

- If there are no two red/blue neighbors, mark all red/blue neighbors

Delete all unmarked vertices

of degree ≥ 3

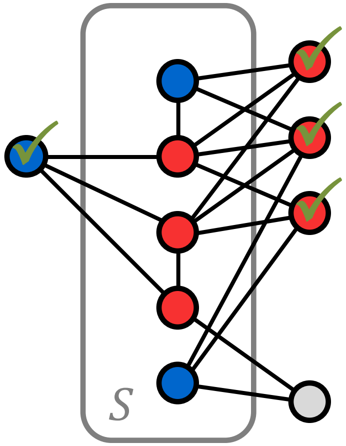


Marking procedure

For every $x \in S$, mark 2 red and 2 blue neighbors in $V(G) \setminus S$

- If there are no two red/blue neighbors, mark all red/blue neighbors

Delete all unmarked vertices of degree ≥ 3



Correctness

Suppose G' is CNCF-colorable

- Color G the same, color removed vertices as they were before removal
  - For $v \notin S$, this conflict-free colors $\mathrm{N}[v]$ by definition
  - For $v \in S$, this never destroys the conflict-free coloring of $N[v]$

Suppose G is CNCF-colorable

- Use the same coloring for G'

Kernel size

We mark at most 4 vertices for each vertex in S

- At most $4k$ high-degree vertices not in S remain

Combined with removing low-degree vertices

- k vertices in S
- $O(k^2)$ degree-≤2 vertices not in S
- $O(k)$ degree-≥3 vertices not in S
- Total of $O(k^2)$ vertices

Theorem

2-CNCF-Coloring Extension parameterized by $|S|$ has a kernel of size $O(k^2 \log k)$, and no kernel[1] of size $O(k^{2-\varepsilon})$

[1] Unless $NP \subseteq coNP/poly$

Polynomial kernel for 2-CNCF-Coloring

Parameterized by Vertex Cover size

A generalized kernel

We give a generalized kernel to d-Polynomial root CSP

Input: A set L of equalities over variables X, where each equality is of the form $p(x_1, \ldots, x_n) = 0$, where p is a polynomial of degree at most d

Parameter: The number of variables n

Question: Does there exist an assignment $\tau: X \to \{0,1\}$ satisfying all equalities in L?

Example of 2-Poly root CSP:

$$\{x_1 + x_2 - 1 = 0, \qquad x_1 * x_2 + x_2 * x_3 = 0\}$$

Satisfied by

$$\tau(x_1) = \tau(x_3) = 0, \tau(x_2) = 1$$

A generalized kernel

We give a generalized kernel to d-Polynomial root CSP

Input: A set L of equalities over variables X, where each equality is of the form $p(x_1, \ldots, x_n) = 0$, where p is a polynomial of degree at most d

Parameter: The number of variables n

Question: Does there exist an assignment $\tau: X \to \{0,1\}$ satisfying all equalities in L?

Theorem [more on this tomorrow, Jansen and Pieterse, MFCS 2016]

d-Polynomial root CSP has a kernel with $O(n^d)$ equalities, where n is the number of variables

- That is a subset of the original set of equalities!

Kernelization: General idea

Three steps

1. Reduce the number of low-degree vertices
2. Rewrite the problem to an instance of d-Poly root CSP
   - For some constant d
   - Using not too many variables
     - Tricky!
3. Apply (known) kernelization result for d-Poly root CSP
   - d-Polynomial root CSP has a kernel with $O(\#vars^d)$ equalities

Removing low-degree vertices

- Same as before: marking procedure to reduce the number of degree-1 and 2 vertices outside S
- Reduces their number to $O(k^2)$
- Add these to S
  - Technically, this increases $|S|$ to $O(k^2)$, but we ignore this for simplicity

Rewriting: Basics

Creating an instance of d-Poly root CSP (for some d)

- For each vertex v, create variables r_v and b_v
  - $r_v = 1$ means v is red, $b_v = 1$ means it is blue
  - Add the constraint that $r_v + b_v = 1$
- A constraint on the coloring of $N[v]$ for all v
  - Exactly one blue, or exactly one red vertex
  - Thus, $\sum_{u \in N[v]} r_u = 1$, or $\sum_{u \in N[v]} b_u = 1$
  - For all v add the constraint
    - $\left(1 - \sum_{u \in N[v]} r_u\right)\left(1 - \sum_{u \in N[v]} b_u\right) = 0$

Rewriting: Continued

So far, variables $\{ r_v, b_v \mid v \in V(G) \}$, constraints

- For all v: $r_v + b_v = 1$
- For all v: $\left(1 - \sum_{u \in N[v]} r_u\right)\left(1 - \sum_{u \in N[v]} b_u\right) = 0$

Hereby

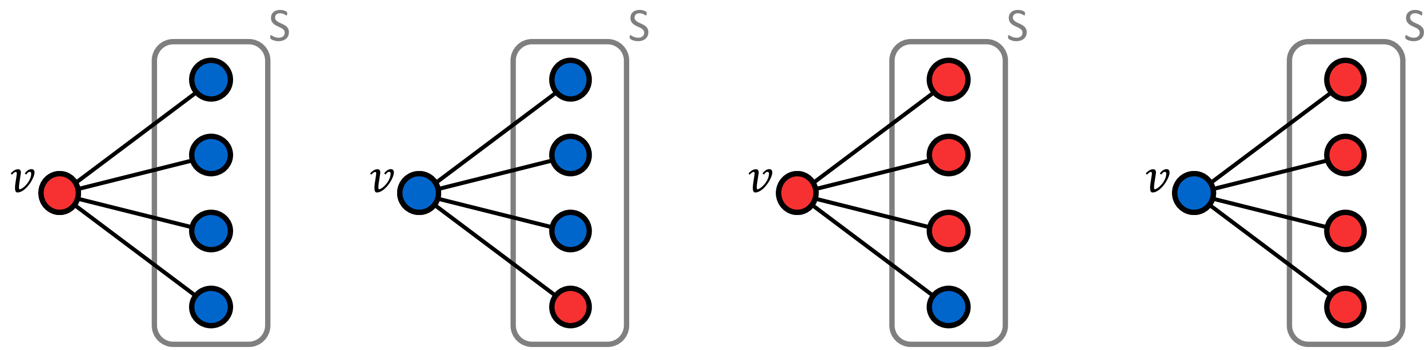
- The two problem instances are equivalent
- We use low-degree polynomials (degree-2)
- As many variables as vertices
  - Using the known kernel for d-Poly root CSP gives a kernel of size $O(n^2)$ (useless)

Plan: reduce the number of variables to $\mathrm{O}(k)$?

Reducing the number of variables

Recall, each vertex $v \notin S$ has degree at least 3

- Its coloring is precisely determined by the colors of $N(v)$



Idea: write r_v as $f\left(r_{u_1}, \dots, r_{u_k}, b_{u_1}, \dots, b_{u_k}\right)$ if $N(v) = \{u_1, \dots, u_k\}$

- For low-degree polynomial f
- Then $b_v = 1 - r_v = 1 - f(\dots)$
- Substituting r_v by $f(\dots)$, reduces the number of variables to $2|S| = 2k$

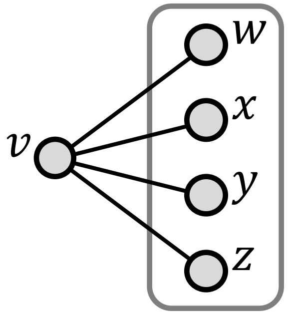
Reducing the number of variables

First, ensure that the neighborhood of $v \notin S$ is "ok"

- All blue, all red, one red, or one blue
- Done by an additional equality of degree 4 for $v \notin S$
  - $(\sum_{u \in N(v)} r_u)(\sum_{u \in N(v)} b_u)(1 - \sum_{u \in N(v)} r_u)(1 - \sum_{u \in N(v)} b_u) = 0$

Defining f

- $r_w + r_x + r_y + r_z = 4$ implies $r_v = 0$
- $r_w + r_x + r_y + r_z = 3$ implies $r_v = 1$
- $r_w + r_x + r_y + r_z = 1$ implies $r_v = 0$
- $r_w + r_x + r_y + r_z = 0$ implies $r_v = 1$



Let g such that $g(x) = 0$ if $x \in \{1,4\}$ and $g(x) = 1$ if $x \in \{0,3\}$

- Substitute $r_v = f(r_w, r_x, r_y, r_z) = g(r_w + r_x + r_y + r_z)$

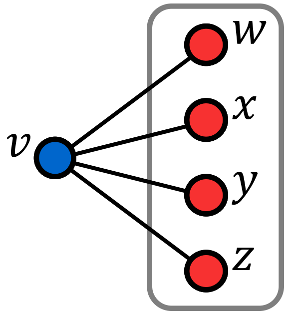
Reducing the number of variables

First, ensure that the neighborhood of $v \notin S$ is "ok"

- All blue, all red, one red, or one blue
- Done by an additional equality of degree 4 for $v \notin S$
  - $(\sum_{u\in N(v)} r_u)(\sum_{u\in N(v)} b_u)(1-\sum_{u\in N(v)} r_u)(1-\sum_{u\in N(v)} b_u) = 0$

Defining f

- $r_w + r_x + r_y + r_z = 4$ implies $r_v = 0$
- $r_w + r_x + r_y + r_z = 3$ implies $r_v = 1$
- $r_w + r_x + r_y + r_z = 1$ implies $r_v = 0$
- $r_w + r_x + r_y + r_z = 0$ implies $r_v = 1$



Let g such that $g(x) = 0$ if $x \in \{1,4\}$ and $g(x) = 1$ if $x \in \{0,3\}$

- Substitute $r_v = f(r_w, r_x, r_y, r_z) = g(r_w + r_x + r_y + r_z)$

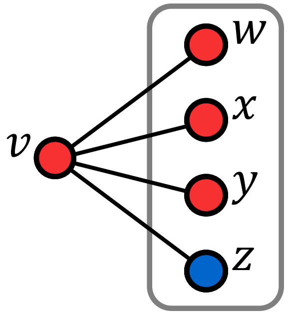
Reducing the number of variables

First, ensure that the neighborhood of $v \notin S$ is "ok"

- All blue, all red, one red, or one blue
- Done by an additional equality of degree 4 for $v \notin S$
  - $(\sum_{u \in N(v)} r_u)(\sum_{u \in N(v)} b_u)(1 - \sum_{u \in N(v)} r_u)(1 - \sum_{u \in N(v)} b_u) = 0$

Defining f

- $r_w + r_x + r_y + r_z = 4$ implies $r_v = 0$
- $r_w + r_x + r_y + r_z = 3$ implies $r_v = 1$
- $r_w + r_x + r_y + r_z = 1$ implies $r_v = 0$
- $r_w + r_x + r_y + r_z = 0$ implies $r_v = 1$



Let g such that $g(x) = 0$ if $x \in \{1,4\}$ and $g(x) = 1$ if $x \in \{0,3\}$

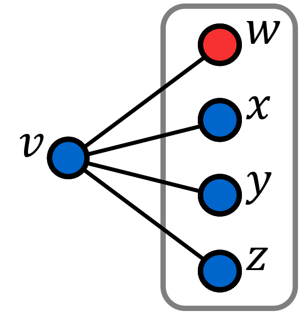
- Substitute $r_v = f(r_w, r_x, r_y, r_z) = g(r_w + r_x + r_y + r_z)$

Reducing the number of variables

First, ensure that the neighborhood of $v \notin S$ is "ok"

- All blue, all red, one red, or one blue
- Done by an additional equality of degree 4 for $v \notin S$
  - $(\sum_{u \in N(v)} r_u)(\sum_{u \in N(v)} b_u)(1 - \sum_{u \in N(v)} r_u)(1 - \sum_{u \in N(v)} b_u) = 0$

Defining f



- $r_w + r_x + r_y + r_z = 4$ implies $r_v = 0$
- $r_w + r_x + r_y + r_z = 3$ implies $r_v = 1$
- $r_w + r_x + r_y + r_z = 1$ implies $r_v = 0$
- $r_w + r_x + r_y + r_z = 0$ implies $r_v = 1$

Let g such that $g(x) = 0$ if $x \in \{1,4\}$ and $g(x) = 1$ if $x \in \{0,3\}$

- Substitute $r_v = f(r_w, r_x, r_y, r_z) = g(r_w + r_x + r_y + r_z)$

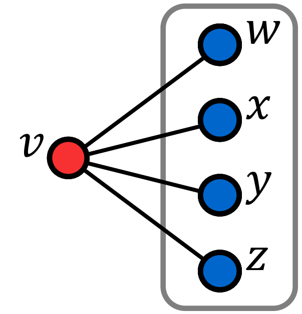
Reducing the number of variables

First, ensure that the neighborhood of $v \notin S$ is "ok"

- All blue, all red, one red, or one blue
- Done by an additional equality of degree 4 for $v \notin S$
  - $(\sum_{u \in N(v)} r_u)(\sum_{u \in N(v)} b_u)(1 - \sum_{u \in N(v)} r_u)(1 - \sum_{u \in N(v)} b_u) = 0$

Defining f

- $r_w + r_x + r_y + r_z = 4$ implies $r_v = 0$
- $r_w + r_x + r_y + r_z = 3$ implies $r_v = 1$
- $r_w + r_x + r_y + r_z = 1$ implies $r_v = 0$
- $r_w + r_x + r_y + r_z = 0$ implies $r_v = 1$



Let g such that $g(x) = 0$ if $x \in \{1,4\}$ and $g(x) = 1$ if $x \in \{0,3\}$

- Substitute $r_v = f(r_w, r_x, r_y, r_z) = g(r_w + r_x + r_y + r_z)$

Reducing the number of variables

In general, for $v \notin S$, find polynomial g s.t.

- $g(x) = 0$ if $x \in \{1, |N(v)|\}$
- $g(x) = 1$ if $x \in \{0, |N(v)| - 1\}$

Use interpolating polynomial for $\{1,0\}, \{|N(v)|, 0\}, \{0,1\}, \{|N(v)| - 1,1\}$

- Has degree 3

Let $N(r_v) = \{u_1, \dots, u_m\}$

- Substitute r_v by $g\left(r_{u_1} + r_{u_2} + \cdots + r_{u_m}\right)$ in all equalities.

Size and correctness

Now: variables $\{ r_v, b_v \mid v \in V(G) \}$, constraints

- For all v: $r_v + b_v = 1$
- For all v: $\left(1 - \sum_{u \in N[v]} r_u\right)\left(1 - \sum_{u \in N[v]} b_u\right) = 0$
  - With for $v \notin S$ r_v substituted by $\mathrm{g}(\dots)$, b_v by $1 - g(\dots)$
- For all $v \notin S$:

$(\sum_{u \in N(v)} r_u)(\sum_{u \in N(v)} b_u)(1 - \sum_{u \in N(v)} r_u)(1 - \sum_{u \in N(v)} b_u) = 0$

Polynomials have degree≤ 6

Replacing r_v by $g(\dots)$ is safe

- One direction, obvious
- Other direction: additional constraint ensures
  - $r_{u_1} + r_{u_2} + \cdots + r_{u_m} \in \{0,1, |N(v)| - 1|, |N(v)|\}$
  - g chosen such that it $g(\dots) = r_v$

Kernelization

We obtained a d-Poly root CSP instance

- $d = 6$
- On $2k$ variables (actually, k variables suffices)

That is equivalent to the original instance

- Apply kernel for 6-Poly root CSP
  - Instance with $O\left(\#vars^d\right) = O(k^6)$ equalities
  - Can be encoded in $O(k^{10})$ bits

Theorem

2-CNCF-Coloring parameterized by Vertex Cover has a generalized kernel of size $O(k^{10})$

- Can be turned into normal kernel of polynomial size

Conclusion

- 2-CNCF-Coloring parameterized by vertex cover has a polynomial kernel
- q-CNCF-Coloring for $q \geq 3$ and q-ONCF-Coloring do not
  - Not even for the extension problem

Open questions

- Is the $O(k^{10})$ bound tight for 2-CNCF-Coloring?
  - Probably not
- Is there an “easier” kernel?

Conclusion

- 2-CNCF-Coloring parameterized by vertex cover has a polynomial kernel
- q-CNCF-Coloring for $q \geq 3$ and q-ONCF-Coloring do not
  - Not even for the extension problem

Open questions

- Is the $O(k^{10})$ bound tight for 2-CNCF-Coloring?
  - Probably not
- Is there an “easier” kernel?

THANK YOU