# Locally checkable problems parameterized by treewidth, clique-width and mim-width

Carolina Lucía Gonzalez[1]

[1]CONICET - Universidad de Buenos Aires, ICC, Buenos Aires, Argentina

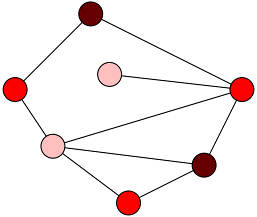
[2]University of Fribourg, Department of Informatics, Fribourg, Switzerland

Joint work with:

- Flavia Bonomo-Braberman[1] (treewidth)
- Narmina Baghirova[2], Bernard Ries[2] and David Schindl[2] (clique-width)
- Felix Mann[2] (mim-width)

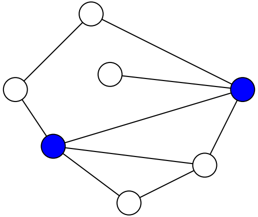
# Introductory problems

**$k$-coloring**



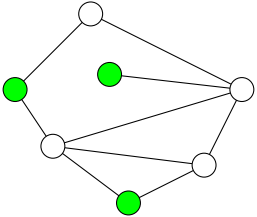
Does there exist a function $c\colon V \to \{1, \ldots, k\}$ such that $c(v) \neq c(u)\ \forall v \in V, u \in N(v)$?

**Minimum Dominating Set**



Minimum size of a set $D$ such that $N[v] \cap D \neq \emptyset$ for all $v \in V$?

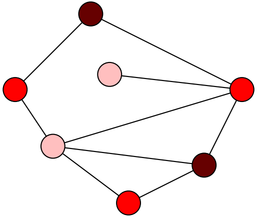
**Maximum Independent Set**



Maximum size of a set $I$ such that $N(v) \cap I = \emptyset$ for all $v \in I$?

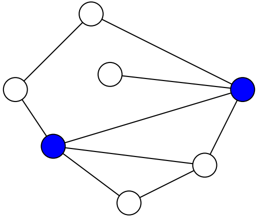
# Introductory problems

## $k$-coloring



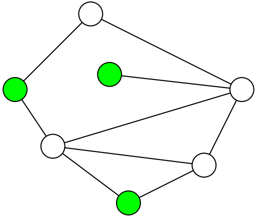
Does there exist a function $c\colon V \to \{1, \ldots, k\}$ such that $c(v) \neq c(u)\ \forall v \in V, u \in N(v)$?

## Minimum Dominating Set



Minimum size of a set $D$ such that $N[v] \cap D \neq \emptyset$ for all $v \in V$?

## Maximum Independent Set

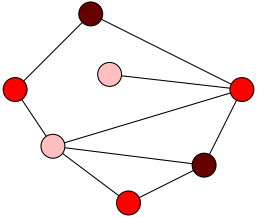


Maximum size of a set $I$ such that $N(v) \cap I = \emptyset$ for all $v \in I$?

What do they have in common?

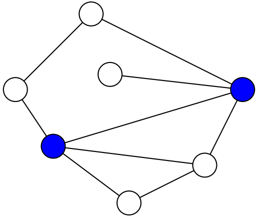
# Introductory problems

### $k$-coloring



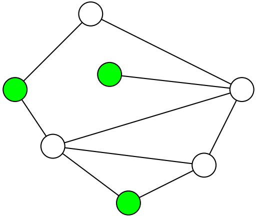
Does there exist a function $c\colon V \to \{1, \ldots, k\}$ such that $c(v) \neq c(u)\ \forall v \in V, u \in N(v)$?

### Minimum Dominating Set



Minimum size of a set $D$ such that $N[v] \cap D \neq \emptyset$ for all $v \in V$?

### Maximum Independent Set



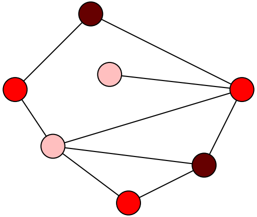
Maximum size of a set $I$ such that $N(v) \cap I = \emptyset$ for all $v \in I$?

What do they have in common?

They are partitioning (coloring) problems where a solution can be verified locally for each vertex.

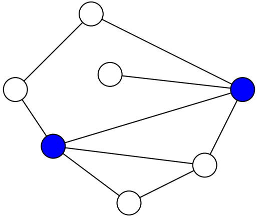
# Introductory problems

**$k$-coloring**



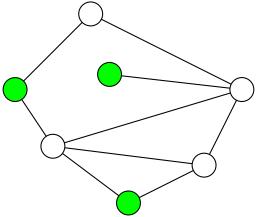
Does there exist a function $c\colon V \to \{1,\ldots,k\}$ such that $c(v) \neq c(u)\ \forall v \in V, u \in N(v)$?

**Minimum Dominating Set**



Minimum size of a set $D$ such that $N[v] \cap D \neq \emptyset$ for all $v \in V$?

**Maximum Independent Set**



Maximum size of a set $I$ such that $N(v) \cap I = \emptyset$ for all $v \in I$?

What do they have in common?

They are partitioning (coloring) problems where a solution can be verified locally for each vertex.

$\rightarrow$ **Locally checkable problems**

# Related work

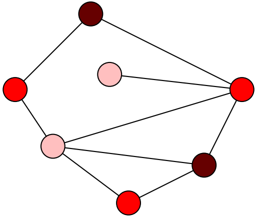
- Monadic Second Order Logic (Courcelle's theorem)
  [B. Courcelle 1990]

- DN-logic
  [B. Bergougnoux, J. Dreier and L. Jaffke 2022]

- Locally Checkable Vertex Partitioning (LCVP) problems
  [J.A. Telle 1994]

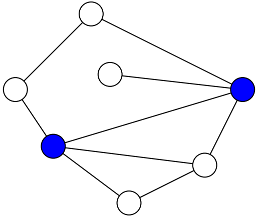
# Introductory problems

**$k$-coloring**



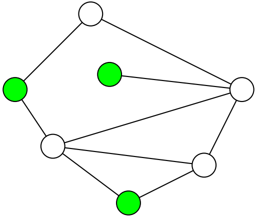
Does there exist a function $c\colon V \to \{1,\dots,k\}$ such that $c(v) \neq c(u)\ \forall v \in V, u \in N(v)$?

**Minimum Dominating Set**



Minimum size of a set $D$ such that $N[v] \cap D \neq \emptyset$ for all $v \in V$?

**Maximum Independent Set**



Maximum size of a set $I$ such that $N(v) \cap I = \emptyset$ for all $v \in I$?

What do they have in common?

They are partitioning (coloring) problems where a solution can be verified locally for each vertex.

$\rightarrow$ **Locally checkable problems**

# Our framework (simplified)

Given a graph $G$ and

- **Colors**: a set of colors,

# Our framework (simplified)

Given a graph $G$ and

- **Colors**: a set of colors,
- $\boldsymbol{check(v, c)}$: a check function (input: vertex $v$ and coloring $c$ of $N[v]$, output: true or false),

# Our framework (simplified)

Given a graph $G$ and

- $\textsc{Colors}$: a set of colors,
- $\boldsymbol{check(v, c)}$: a check function (input: vertex $v$ and coloring $c$ of $N[v]$, output: true or false),
- $\mathbf{w}(\boldsymbol{v}, \boldsymbol{c})$: a weight function (input: vertex $v$ and coloring $c$ of $N[v]$, output: a weight),

# Our framework (simplified)

Given a graph $G$ and

- **Colors**: a set of colors,
- $\boldsymbol{check(v, c)}$: a check function (input: vertex $v$ and coloring $c$ of $N[v]$, output: true or false),
- $\mathbf{w}(\boldsymbol{v}, \boldsymbol{c})$: a weight function (input: vertex $v$ and coloring $c$ of $N[v]$, output: a weight),
- an **order** of the weights

# Our framework (simplified)

Given a graph $G$ and

- $\textsc{Colors}$: a set of colors,
- $\boldsymbol{check(v, c)}$: a check function (input: vertex $v$ and coloring $c$ of $N[v]$, output: true or false),
- $\mathbf{w}(\boldsymbol{v, c})$: a weight function (input: vertex $v$ and coloring $c$ of $N[v]$, output: a weight),
- an **order** of the weights

find the minimum weight of a coloring $c$ such that

$$check(v, c|_{N[v]}) = \textsc{True} \quad \forall\, v \in V(G).$$

# Our framework (simplified)

Given a graph $G$ and

- **$\textsc{Colors}$**: a set of colors,
- ***check*$(v,c)$**: a check function (input: vertex $v$ and coloring $c$ of $N[v]$, output: true or false),
- **$\mathrm{W}(v,c)$**: a weight function (input: vertex $v$ and coloring $c$ of $N[v]$, output: a weight),
- an **order** of the weights

find the minimum weight of a coloring $c$ such that

$$check(v, c|_{N[v]}) = \textsc{True} \quad \forall v \in V(G).$$

Minimum Dominating Set:

$\textsc{Colors} = \{\mathbf{s}, \overline{\mathbf{s}}\}$

$\mathrm{W}(v,c) = 1$ if $c(v) = \mathbf{s}$, and $0$ otherwise

Order of weights: $\leq$

$check(v,c) = (c(v) = \mathbf{s} \vee \exists u \in N(v).\, c(u) = \mathbf{s})$

# Our framework (simplified)

Given a graph $G$ and

- $\textsc{Colors}$: a set of colors,
- $\boldsymbol{check(v,c)}$: a check function (input: vertex $v$ and coloring $c$ of $N[v]$, output: true or false),
- $\mathbf{w}(\boldsymbol{v},\boldsymbol{c})$: a weight function (input: vertex $v$ and coloring $c$ of $N[v]$, output: a weight),
- an **order** of the weights

find the minimum weight of a coloring $c$ such that

$$check(v, c|_{N[v]}) = \textsc{True} \quad \forall\, v \in V(G).$$

Goal: obtain efficient algorithms for different graph classes.

# Our framework (simplified)

Given a graph $G$ and

- **COLORS**: a set of colors,
- $\boldsymbol{check(v, c)}$: a check function (input: vertex $v$ and coloring $c$ of $N[v]$, output: true or false),
- $\mathbf{w}(\boldsymbol{v}, \boldsymbol{c})$: a weight function (input: vertex $v$ and coloring $c$ of $N[v]$, output: a weight),
- an **order** of the weights

find the minimum weight of a coloring $c$ such that

$$check(v, c|_{N[v]}) = \textsc{True} \quad \forall\, v \in V(G).$$

Goal: obtain efficient algorithms for different graph classes. $\longrightarrow$ **Under which conditions?**

# Our framework (simplified)

Given a graph $G$ and

- $\textsc{Colors}$: a set of colors,
- $\mathit{check}(v, c)$: a check function (input: vertex $v$ and coloring $c$ of $N[v]$, output: true or false),
- $\mathrm{w}(v, c)$: a weight function (input: vertex $v$ and coloring $c$ of $N[v]$, output: a weight),
- an **order** of the weights

find the minimum weight of a coloring $c$ such that

$$\mathit{check}(v, c|_{N[v]}) = \textsc{True} \quad \forall\, v \in V(G).$$

Goal: obtain efficient algorithms for different graph classes. $\rightarrow$ **Under which conditions?**

In particular, we focus on the parameterization by different width measures:

- treewidth
- clique-width
- mim-width

# Our framework (simplified)

Given a graph $G$ and

- $\textsc{Colors}$: a set of colors,
- $check(v, c)$: a check function (input: vertex $v$ and coloring $c$ of $N[v]$, output: true or false),
- $\mathrm{w}(v, c)$: a weight function (input: vertex $v$ and coloring $c$ of $N[v]$, output: a weight),
- an **order** of the weights

find the minimum weight of a coloring $c$ such that

$$check(v, c|_{N[v]}) = \textsc{True} \quad \forall v \in V(G).$$

Goal: obtain efficient algorithms for different graph classes. $\rightarrow$ **Under which conditions?**

In particular, we focus on the parameterization by different width measures:

- ~~treewidth~~ $\rightarrow$ not today...
- clique-width
- mim-width

# Some conditions on $check$ and $\mathrm{W}$

Let $\textsc{Colors} = \{a_1, \ldots, a_q\}$ be a set of colors.

# Some conditions on *check* and W

Let $\textsc{Colors} = \{a_1, \ldots, a_q\}$ be a set of colors.

A function $f$ is **color-counting** if there exists $f'$ such that

$$f(v, c) = f'(v, c(v), |N^c_{a_1}(v)|, \ldots, |N^c_{a_q}(v)|)$$

for all vertex $v$, coloring $c$ of $N[v]$.

# Some conditions on $check$ and $\mathrm{W}$

Let $\textsc{Colors} = \{a_1, \dots, a_q\}$ be a set of colors.

A function $f$ is **color-counting** if there exists $f'$ such that

$$f(v, c) = f'(v, c(v), |N^c_{a_1}(v)|, \dots, |N^c_{a_q}(v)|)$$

for all vertex $v$, coloring $c$ of $N[v]$.

Informally: ...if $f$ only depends on the vertex, the color it receives and the number of neighbors of each color ($\Rightarrow$ existence of $f'(v, a, n_1, \dots, n_q)$)

# Some conditions on $check$ and W

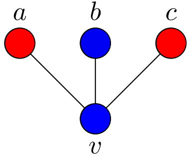
Let $\text{Colors} = \{a_1, \ldots, a_q\}$ be a set of colors.

A function $f$ is **color-counting** if there exists $f'$ such that

$$f(v, c) = f'(v, c(v), |N^c_{a_1}(v)|, \ldots, |N^c_{a_q}(v)|)$$

for all vertex $v$, coloring $c$ of $N[v]$.

Informally: ...if $f$ only depends on the vertex, the color it receives and the number of neighbors of each color ($\Rightarrow$ existence of $f'(v, a, n_1, \ldots, n_q)$)



# Some conditions on $\mathit{check}$ and $\mathrm{W}$

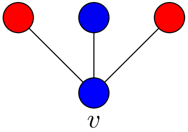
Let $\textsc{Colors} = \{a_1, \ldots, a_q\}$ be a set of colors.

A function $f$ is **color-counting** if there exists $f'$ such that

$$f(v, c) = f'(v, c(v), |N^c_{a_1}(v)|, \ldots, |N^c_{a_q}(v)|)$$

for all vertex $v$, coloring $c$ of $N[v]$.

Informally: ...if $f$ only depends on the vertex, the color it receives and the number of neighbors of each color ($\Rightarrow$ existence of $f'(v, a, n_1, \ldots, n_q)$)




$v$ is blue, 2 red neighbors, 1 blue neighbor

# Some conditions on $\mathit{check}$ and W

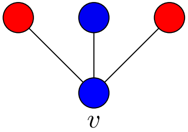
Let $\textsc{Colors} = \{a_1, \ldots, a_q\}$ be a set of colors.

A function $f$ is **color-counting** if there exists $f'$ such that

$$f(v, c) = f'(v, c(v), |N^c_{a_1}(v)|, \ldots, |N^c_{a_q}(v)|)$$

for all vertex $v$, coloring $c$ of $N[v]$.

Informally: ...if $f$ only depends on the vertex, the color it receives and the number of neighbors of each color ($\Rightarrow$ existence of $f'(v, a, n_1, \ldots, n_q)$)




$v$ is blue, 2 red neighbors, 1 blue neighbor

$$f'(v, \text{B}, 2, 1)$$

# Some conditions on $check$ and $\mathrm{W}$

Let $\textsc{Colors} = \{a_1, \ldots, a_q\}$ be a set of colors.

A function $f$ is **color-counting** if there exists $f'$ such that

$$f(v, c) = f'(v, c(v), |N^c_{a_1}(v)|, \ldots, |N^c_{a_q}(v)|)$$

for all vertex $v$, coloring $c$ of $N[v]$.

Informally: ...if $f$ only depends on the vertex, the color it receives and the number of neighbors of each color ($\Rightarrow$ existence of $f'(v, a, n_1, \ldots, n_q)$)

Example: minimum dominating set
$check(v, c) = (c(v) = \mathbf{s} \vee \exists u \in N(v).\, c(u) = \mathbf{s})$

# Some conditions on $check$ and $\mathrm{W}$

Let $\textsc{Colors} = \{a_1, \ldots, a_q\}$ be a set of colors.

A function $f$ is **color-counting** if there exists $f'$ such that

$$f(v,c) = f'(v, c(v), |N^c_{a_1}(v)|, \ldots, |N^c_{a_q}(v)|)$$

for all vertex $v$, coloring $c$ of $N[v]$.

Informally: ...if $f$ only depends on the vertex, the color it receives and the number of neighbors of each color ($\Rightarrow$ existence of $f'(v, a, n_1, \ldots, n_q)$)

Example: minimum dominating set

$check(v,c) = (c(v) = \mathbf{s} \vee \exists u \in N(v).\, c(u) = \mathbf{s})$

$check'(v, a, n_{\mathbf{s}}, n_{\overline{\mathbf{s}}}) = (a = \mathbf{s} \vee n_{\mathbf{s}} \geq 1)$

# Some conditions on $check$ and W

Let $\textsc{Colors} = \{a_1, \ldots, a_q\}$ be a set of colors.

A function $f$ is **color-counting** if there exists $f'$ such that

$$f(v, c) = f'(v, c(v), |N^c_{a_1}(v)|, \ldots, |N^c_{a_q}(v)|)$$

for all vertex $v$, coloring $c$ of $N[v]$.

Informally: ...if $f$ only depends on the vertex, the color it receives and the number of neighbors of each color ($\Rightarrow$ existence of $f'(v, a, n_1, \ldots, n_q)$)

Example: minimum dominating set
$check(v, c) = (c(v) = \mathbf{s} \vee \exists u \in N(v).\, c(u) = \mathbf{s})$
$check'(v, a, n_{\mathbf{s}}, n_{\overline{\mathbf{s}}}) = (a = \mathbf{s} \vee n_{\mathbf{s}} \geq 1)$

We say $f$ is ***d*-stable** if it is color-counting and

$$f'(v, a, n_1, \ldots, n_q) = f'(v, a, \min(d, n_1), \ldots, \min(d, n_q))$$

for all vertex $v$, color $a$, non-negative integers $n_1, \ldots, n_q$.

# Some conditions on $\mathit{check}$ and $\mathrm{W}$

Let $\textsc{Colors} = \{a_1, \ldots, a_q\}$ be a set of colors.

A function $f$ is **color-counting** if there exists $f'$ such that

$$f(v,c) = f'(v, c(v), |N^c_{a_1}(v)|, \ldots, |N^c_{a_q}(v)|)$$

for all vertex $v$, coloring $c$ of $N[v]$.

Informally: ...if $f$ only depends on the vertex, the color it receives and the number of neighbors of each color ($\Rightarrow$ existence of $f'(v, a, n_1, \ldots, n_q)$)

Example: minimum dominating set
$\mathit{check}(v,c) = (c(v) = \mathbf{s} \vee \exists u \in N(v).\, c(u) = \mathbf{s})$
$\mathit{check}'(v, a, n_{\mathbf{s}}, n_{\overline{\mathbf{s}}}) = (a = \mathbf{s} \vee n_{\mathbf{s}} \geq 1)$ $\rightarrow$ **1-stable**

We say $f$ is ***d*-stable** if it is color-counting and

$$f'(v, a, n_1, \ldots, n_q) = f'(v, a, \min(d, n_1), \ldots, \min(d, n_q))$$

for all vertex $v$, color $a$, non-negative integers $n_1, \ldots, n_q$.

# Complexity

- If $|\textsc{Colors}|$ is not a constant:

# Complexity

- If $|\textsc{Colors}|$ is not a constant:

**Precoloring Extension** is **NP-complete** on graphs of clique-width $\leq 3$.

[F. Bonomo, G. Durán and J. Marenco 2006]

# Complexity

- If $|\textsc{Colors}|$ is not a constant: para-NP-hard clique-width (even for 1-stable $check$ and $\mathrm{W}$)

**Precoloring Extension** is **NP-complete** on graphs of clique-width $\leq 3$.

[F. Bonomo, G. Durán and J. Marenco 2006]

# Complexity

- If $|\textsc{Colors}|$ is not a constant: para-NP-hard clique-width (even for 1-stable $check$ and $\textsc{w}$)

- If $|\textsc{Colors}|$ is a constant:

| $check, \textsc{w}$ | Clique-width | Mim-width |
|---|---|---|
| Non color-counting | para-NP-hard | |
| Color-counting | XP | para-NP-hard |
| $d$-stable | FPT | XP |

# Complexity

- If $|\textsc{Colors}|$ is not a constant: para-NP-hard clique-width (even for 1-stable *check* and $\textsc{w}$)

- If $|\textsc{Colors}|$ is a constant:

| $check, \textsc{w}$ | Clique-width | Mim-width |
|---|---|---|
| Non color-counting | para-NP-hard | |
| Color-counting | XP | para-NP-hard |
| $d$-stable | FPT | XP |

We can reduce Minimum Dominating Set in general graphs to a locally checkable problem (with 2 colors) in complete graphs (clique-width $\leq 2$).

# Complexity

- If $|\textsc{Colors}|$ is not a constant: para-NP-hard clique-width (even for 1-stable $check$ and $\textsc{w}$)

- If $|\textsc{Colors}|$ is a constant:

| $check, \textsc{w}$ | Clique-width | Mim-width |
|---|---|---|
| Non color-counting | para-NP-hard | |
| Color-counting | XP | para-NP-hard |
| $d$-stable | FPT | XP |

Standard dynamic programming algorithm using clique-width expressions.

# Complexity

- If $|\textsc{Colors}|$ is not a constant: para-NP-hard clique-width (even for 1-stable $check$ and $\textsc{w}$)

- If $|\textsc{Colors}|$ is a constant:

| $check, \textsc{w}$ | Clique-width | Mim-width |
|---|---|---|
| Non color-counting | para-NP-hard | |
| Color-counting | XP | para-NP-hard |
| $d$-stable | FPT | XP |

**Max-Cut** is color-counting with 2 colors, and **W[1]-hard** parameterized by clique-width.

[F.V. Fomin, P.A. Golovach, D. Lokshtanov and S. Saurabh 2014]

# Complexity

- If $|\textsc{Colors}|$ is not a constant: para-NP-hard clique-width (even for 1-stable $check$ and $\textsc{w}$)

- If $|\textsc{Colors}|$ is a constant:

| $check, \textsc{w}$ | Clique-width | Mim-width |
|---|---|---|
| Non color-counting | para-NP-hard | |
| Color-counting | XP | para-NP-hard |
| $d$-stable | FPT | XP |

Dynamic programming algorithm based on [B.M. Bui-Xuan, J.A. Telle and M. Vatshelle 2013].

Alternative: modeling with DN-logic.

# Complexity

- If $|\textsc{Colors}|$ is not a constant: para-NP-hard clique-width (even for 1-stable $check$ and $\textsc{w}$)

- If $|\textsc{Colors}|$ is a constant:

| $check, \textsc{w}$ | Clique-width | Mim-width |
|---|---|---|
| Non color-counting | para-NP-hard | |
| Color-counting | XP | para-NP-hard |
| $d$-stable | FPT | XP |

**Minimum Dominating Set** is 1-stable with 2 colors, and **W[1]-hard** parameterized by mim-width.
[F.V. Fomin, P.A. Golovach, J.F. Raymond 2018]

# Complexity

- If $|\textsc{Colors}|$ is not a constant: para-NP-hard clique-width (even for 1-stable *check* and $\textsc{w}$)

- If $|\textsc{Colors}|$ is a constant:

| $check, \textsc{w}$ | Clique-width | Mim-width |
|---|---|---|
| Non color-counting | para-NP-hard | |
| Color-counting | XP | para-NP-hard |
| $d$-stable | FPT | XP |

**Max-Cut** is color-counting with 2 colors, and **NP-complete** on interval graphs (mim-width $\leq 1$).

[R. Adhikary, K. Bose, S. Mukherjee, B. Roy 2020]

# Adding global properties

- Treewidth: size, connectivity, acyclicity
- Clique-width: size, connectivity
- Mim-width: size, connectivity and any other expressible in DN-logic

# Applications

Using this framework we proved that:

| **Problem** | **clique-width** | **mim-width** |
|---|---|---|
| $[k]$-Roman domination | (linear$^*$) FPT | XP |
| Conflict-free $k$-coloring<br>Bhyravarapu, Hartmann, Kalyanasundaram and Vinod Reddy, 2021: similar results for clique-width | (linear$^*$) FPT | XP |
| b-coloring with fixed number of colors<br>Jaffke, Lima and Lokshtanov, 2020: XP parameterized by clique-width with unfixed number of colors | (linear$^*$) FPT | XP |
| $k$-community | XP | — |

and similar results for some variations of these problems.

(*) if a clique-width expression is given as input.

# Applications: $[k]$-Roman domination

Given a graph $G$, compute the minimum weight of a function $f\colon V \to \{0, \ldots, k+1\}$ such that

$$f(v) + \sum_{\substack{u \in N(v) \\ f(u) \geq 1}} (f(u) - 1) \geq k \quad \forall\, v \in V.$$

# Applications: $[k]$-Roman domination

Given a graph $G$, compute the minimum weight of a function $f\colon V \to \{0,\dots,k+1\}$ such that

$$f(v) + \sum_{\substack{u \in N(v)\\ f(u) \geq 1}} (f(u) - 1) \geq k \quad \forall\, v \in V.$$

- $\textsc{Colors} = \{0,\dots,k+1\}$
- $\mathrm{w}(v,c) = c(v)$
- Order of weights: $\leq$
- $check(v,c) = \left( c(v) + \sum_{\substack{u \in N(v)\\ c(u) \geq 1}} (c(u) - 1) \geq k \right)$

# Applications: $[k]$-Roman domination

Given a graph $G$, compute the minimum weight of a function $f\colon V \to \{0,\dots,k+1\}$ such that

$$f(v) + \sum_{\substack{u \in N(v)\\ f(u) \geq 1}} (f(u) - 1) \geq k \quad \forall\, v \in V.$$

- **$\textsc{Colors} = \{0,\dots,k+1\}$**
- $\mathrm{w}(v,c) = c(v)$
- Order of weights: $\leq$
- $check(v,c) = \left( c(v) + \sum_{\substack{u \in N(v)\\ c(u) \geq 1}} (c(u) - 1) \geq k \right)$

# Applications: $[k]$-Roman domination

Given a graph $G$, compute the minimum weight of a function $f\colon V \to \{0,\dots,k+1\}$ such that

$$f(v) + \sum_{\substack{u \in N(v)\\ f(u)\geq 1}} (f(u)-1) \geq k \quad \forall\, v \in V.$$

- $\textbf{C}\small{\textbf{OLORS}}\boldsymbol{= \{0,\dots,k+1\}}$
- $\mathrm{W}(v,c) = c(v)$
- Order of weights: $\leq$
- $check(v,c) = \left( c(v) + \sum_{\substack{u \in N(v)\\ c(u)\geq 1}} (c(u)-1) \geq k \right)$

  ***check* and W are color-counting**

# Applications: $[k]$-Roman domination

Given a graph $G$, compute the minimum weight of a function $f\colon V \to \{0, \ldots, k+1\}$ such that

$$f(v) + \sum_{\substack{u \in N(v) \\ f(u) \geq 1}} (f(u) - 1) \geq k \quad \forall\, v \in V.$$

- **$\textsc{Colors} = \{0, \ldots, k+1\}$**
- $\mathrm{w}'(v, i, n_0, \ldots, n_{k+1}) = i$
- Order of weights: $\leq$
- $check(v, c) = \left( c(v) + \sum_{\substack{u \in N(v) \\ c(u) \geq 1}} (c(u) - 1) \geq k \right)$

  **$check$ and $\mathrm{w}$ are color-counting**

# Applications: $[k]$-Roman domination

Given a graph $G$, compute the minimum weight of a function $f\colon V \to \{0,\dots,k+1\}$ such that

$$f(v) + \sum_{\substack{u \in N(v)\\ f(u)\geq 1}} (f(u)-1) \geq k \quad \forall\, v \in V.$$

- $\textsc{Colors} = \{0,\dots,k+1\}$
- $\mathrm{W}'(v,i,n_0,\dots,n_{k+1}) = i$
- Order of weights: $\leq$
- $check'(v,i,n_0,\dots,n_{k+1}) = \left(i + \sum_{j=1}^{k+1}(j-1)\cdot n_j \geq k\right)$

**$check$ and $\mathrm{W}$ are color-counting**

# Applications: $[k]$-Roman domination

Given a graph $G$, compute the minimum weight of a function $f\colon V \to \{0,\dots,k+1\}$ such that

$$f(v) + \sum_{\substack{u \in N(v)\\ f(u)\geq 1}} (f(u)-1) \geq k \quad \forall\, v \in V.$$

- $\textbf{C}\text{OLORS}= \{0,\dots,k+1\}$
- $\mathrm{w}'(v,i,n_0,\dots,n_{k+1}) = i$
- Order of weights: $\leq$
- $check'(v,i,n_0,\dots,n_{k+1}) = \left( i + \sum_{j=1}^{k+1} (j-1)\cdot n_j \geq k \right)$

  ***check* and w are *k*-stable**

# Applications: $[k]$-Roman domination

Given a graph $G$, compute the minimum weight of a function $f\colon V \to \{0, \ldots, k+1\}$ such that

$$f(v) + \sum_{\substack{u \in N(v) \\ f(u) \geq 1}} (f(u) - 1) \geq k \quad \forall\, v \in V.$$

- **$\textsc{Colors} = \{0, \ldots, k+1\}$**
- $\mathrm{W}'(v, i, n_0, \ldots, n_{k+1}) = i$
- Order of weights: $\leq$
- $check'(v, i, n_0, \ldots, n_{k+1}) = \left( i + \sum_{j=1}^{k+1} (j-1) \cdot n_j \geq k \right)$

  ***check* and $\mathrm{W}$ are *$k$*-stable**

→ **FPT clique-width**
→ **XP mim-width**

# Applications: $k$-community

A **community structure** of a graph $G$ is a partition $\{C_1, \ldots, C_k\}$, with $k \geq 2$, of $V$ such that for each $i \in \{1, \ldots, k\}$ we have $|C_i| \geq 2$ and

$$\frac{|N(v) \cap C_i|}{|C_i| - 1} \geq \frac{|N(v) \cap C_j|}{|C_j|} \quad \forall v \in C_i,\, \forall j \in \{1, \ldots, k\}.$$

# Applications: $k$-community

A **community structure** of a graph $G$ is a partition $\{C_1, \ldots, C_k\}$, with $k \geq 2$, of $V$ such that for each $i \in \{1, \ldots, k\}$ we have $|C_i| \geq 2$ and

$$\frac{|N(v) \cap C_i|}{|C_i| - 1} \geq \frac{|N(v) \cap C_j|}{|C_j|} \quad \forall v \in C_i, \forall j \in \{1, \ldots, k\}.$$

## $k$-community problem

Decide if a given graph has a community structure with $k$ communities.

# Applications: $k$-community

A **community structure** of a graph $G$ is a partition $\{C_1, \ldots, C_k\}$, with $k \geq 2$, of $V$ such that for each $i \in \{1, \ldots, k\}$ we have $|C_i| \geq 2$ and

$$\frac{|N(v) \cap C_i|}{\boldsymbol{|C_i|} - 1} \geq \frac{|N(v) \cap C_j|}{\boldsymbol{|C_j|}} \quad \forall v \in C_i, \forall j \in \{1, \ldots, k\}.$$

## $k$-community problem

Decide if a given graph has a community structure with $k$ communities.

$\rightarrow$ **Not entirely locally checkable...**

# Applications: $k$-community

A **community structure** of a graph $G$ is a partition $\{C_1, \ldots, C_k\}$, with $k \geq 2$, of $V$ such that for each $i \in \{1, \ldots, k\}$ we have $|C_i| \geq 2$ and

$$\frac{|N(v) \cap C_i|}{|C_i| - 1} \geq \frac{|N(v) \cap C_j|}{|C_j|} \quad \forall v \in C_i, \forall j \in \{1, \ldots, k\}.$$

## Specified size $k$-community problem

Given a graph $G$ and $k$ integers $s_1, \ldots, s_k \geq 2$, determine if $G$ admits a community structure $\{C_1, \ldots, C_k\}$ such that $|C_i| = s_i \ \forall i \in \{1, \ldots, k\}$.

# Applications: $k$-community

A **community structure** of a graph $G$ is a partition $\{C_1, \ldots, C_k\}$, with $k \geq 2$, of $V$ such that for each $i \in \{1, \ldots, k\}$ we have $|C_i| \geq 2$ and

$$\frac{|N(v) \cap C_i|}{|C_i| - 1} \geq \frac{|N(v) \cap C_j|}{|C_j|} \quad \forall v \in C_i, \forall j \in \{1, \ldots, k\}.$$

## Specified size $k$-community problem

Given a graph $G$ and $k$ integers $s_1, \ldots, s_k \geq 2$, determine if $G$ admits a community structure $\{C_1, \ldots, C_k\}$ such that $|C_i| = s_i \ \forall i \in \{1, \ldots, k\}$.

- $\textsc{Colors} = \{1, \ldots, k\}$
- $check(v, c) = \left( \forall j \in \{1, \ldots, k\}. \ \frac{|N(v) \cap C_{c(v)}|}{s_{c(v)} - 1} \geq \frac{|N(v) \cap C_j|}{s_j} \right)$
- for each color $i$, the size of the color class of $i$ has to be $s_i$.

# Applications: $k$-community

A **community structure** of a graph $G$ is a partition $\{C_1, \ldots, C_k\}$, with $k \geq 2$, of $V$ such that for each $i \in \{1, \ldots, k\}$ we have $|C_i| \geq 2$ and

$$\frac{|N(v) \cap C_i|}{|C_i| - 1} \geq \frac{|N(v) \cap C_j|}{|C_j|} \quad \forall v \in C_i,\ \forall j \in \{1, \ldots, k\}.$$

## Specified size $k$-community problem

Given a graph $G$ and $k$ integers $s_1, \ldots, s_k \geq 2$, determine if $G$ admits a community structure $\{C_1, \ldots, C_k\}$ such that $|C_i| = s_i\ \forall i \in \{1, \ldots, k\}$.

- $\textsc{Colors} = \{1, \ldots, k\}$
- $check'(v, i, n_1, \ldots, n_k) = \left(\forall j \in \{1, \ldots, k\}.\ \dfrac{n_i}{s_i - 1} \geq \dfrac{n_j}{s_j}\right)$
- for each color $i$, the size of the color class of $i$ has to be $s_i$.

# Applications: $k$-community

A **community structure** of a graph $G$ is a partition $\{C_1, \ldots, C_k\}$, with $k \geq 2$, of $V$ such that for each $i \in \{1, \ldots, k\}$ we have $|C_i| \geq 2$ and

$$\frac{|N(v) \cap C_i|}{|C_i| - 1} \geq \frac{|N(v) \cap C_j|}{|C_j|} \quad \forall v \in C_i, \forall j \in \{1, \ldots, k\}.$$

## Specified size $k$-community problem

Given a graph $G$ and $k$ integers $s_1, \ldots, s_k \geq 2$, determine if $G$ admits a community structure $\{C_1, \ldots, C_k\}$ such that $|C_i| = s_i \ \forall i \in \{1, \ldots, k\}$.

- $\textsc{Colors} = \{1, \ldots, k\}$
- $check'(v, i, n_1, \ldots, n_k) = \left(\forall j \in \{1, \ldots, k\}. \, \frac{n_i}{s_i - 1} \geq \frac{n_j}{s_j}\right)$
- for each color $i$, the size of the color class of $i$ has to be $s_i$.

$\rightarrow$ **XP clique-width**

# Applications: $k$-community

A **community structure** of a graph $G$ is a partition $\{C_1, \ldots, C_k\}$, with $k \geq 2$, of $V$ such that for each $i \in \{1, \ldots, k\}$ we have $|C_i| \geq 2$ and

$$\frac{|N(v) \cap C_i|}{|C_i| - 1} \geq \frac{|N(v) \cap C_j|}{|C_j|} \quad \forall v \in C_i, \forall j \in \{1, \ldots, k\}.$$

## $k$-community problem

Decide if a given graph has a community structure with $k$ communities.

# Applications: $k$-community

A **community structure** of a graph $G$ is a partition $\{C_1, \dots, C_k\}$, with $k \geq 2$, of $V$ such that for each $i \in \{1, \dots, k\}$ we have $|C_i| \geq 2$ and

$$\frac{|N(v) \cap C_i|}{|C_i| - 1} \geq \frac{|N(v) \cap C_j|}{|C_j|} \quad \forall v \in C_i,\ \forall j \in \{1, \dots, k\}.$$

## $k$-community problem

Decide if a given graph has a community structure with $k$ communities.

For all $s_1, \dots, s_k$ such that $\sum_{i=1}^{k} s_i = |V|$ and $s_i \geq 2\ \forall i \in \{1, \dots, k\}$, solve the corresponding specified size $k$-community problem.

# Applications: $k$-community

A **community structure** of a graph $G$ is a partition $\{C_1, \ldots, C_k\}$, with $k \geq 2$, of $V$ such that for each $i \in \{1, \ldots, k\}$ we have $|C_i| \geq 2$ and

$$\frac{|N(v) \cap C_i|}{|C_i| - 1} \geq \frac{|N(v) \cap C_j|}{|C_j|} \quad \forall v \in C_i,\ \forall j \in \{1, \ldots, k\}.$$

## $k$-community problem

Decide if a given graph has a community structure with $k$ communities.

For all $s_1, \ldots, s_k$ such that $\sum_{i=1}^{k} s_i = |V|$ and $s_i \geq 2\ \forall i \in \{1, \ldots, k\}$, solve the corresponding specified size $k$-community problem.

$\rightarrow$ **XP clique-width**

# Our framework (complete formulation)

## $r$-locally checkable problems

Given a graph $G$ and

- $\textsc{Colors}$: a set of colors,
- $L_v$: for every vertex $v$, a subset of $\textsc{Colors}$ of allowed colors,
- $\ell_e$: for every edge $e$, a label,
- $(\textsc{Weights}, \preceq, \oplus)$: a weight set,
- $\mathrm{w}(v, c)$: a weight function (input: vertex $v$ and coloring $c$ of $N^r[v]$, output: a weight),
- $check(v, c)$: a check function (input: vertex $v$ and coloring $c$ of $N^r[v]$, output: true or false)

find the minimum weight of a coloring $c$ such that $check(v, c|_{N^r[v]}) = \textsc{True}\ \forall\, v \in V(G)$.

| Width | $\lvert\textsc{Colors}\rvert$ | $\ell_e$ | $check, \mathrm{w}$ | Global properties |
|---|---|---|---|---|
| tw | Polynomial | Yes | Polynomial partial neighborhood system | Size, connectivity, acyclicity |
| cw | Constant or log | No | Color-counting | Size, connectivity |
| mimw | Constant | No | $d$-stable | Size, connectivity and any other expressible in DN-logic |