Locally Checkable Problems Parameterized by Clique-Width

Narmina Baghirova
Department of Informatics, University of Fribourg, Switzerland

Carolina Lucía Gonzalez[1]
Instituto de Investigación en Ciencias de la Computación (ICC),
CONICET-University of Buenos Aires, Argentina

Bernard Ries
Department of Informatics, University of Fribourg, Switzerland

David Schindl
Department of Informatics, University of Fribourg, Switzerland

Abstract

We continue the study initiated by Bonomo-Braberman and Gonzalez in 2020 on r-locally checkable problems. We propose a dynamic programming algorithm that takes as input a graph with an associated clique-width expression and solves a 1-locally checkable problem under certain restrictions. We show that it runs in polynomial time in graphs of bounded clique-width, when the number of colors of the locally checkable problem is fixed. Furthermore, we present a first extension of our framework to global properties by taking into account the sizes of the color classes, and consequently enlarge the set of problems solvable in polynomial time with our approach in graphs of bounded clique-width. As examples, we apply this setting to show that, when parameterized by clique-width, the $[k]-$Roman domination problem is FPT, and the k-community problem, Max PDS and other variants are XP.

2012 ACM Subject Classification Mathematics of computing → Graph algorithms; Mathematics of computing → Graph coloring; Mathematics of computing → Combinatorial optimization; Mathematics of computing → Combinatorial algorithms; Theory of computation → Problems, reductions and completeness

Keywords and phrases locally checkable problem, clique-width, dynamic programming, coloring

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2022.31

Related Version *Full Version*: `https://arxiv.org/abs/2203.02992`

Funding *Carolina Lucía Gonzalez*: This research was conducted during two research stays in the University of Fribourg financed by Universidad de Buenos Aires and University of Fribourg. CONICET PIP 11220200100084CO and UBACyT 20020170100495BA.

1 Introduction

Many graph problems can be stated as a sort of partitioning, or equivalently, as a sort of coloring problem. Furthermore, most decision problems on graphs from the literature belong to the class NP, and their certificate verification algorithms often consist in checking some *local* property for each vertex, i.e. involving itself and its neighborhood only, plus possibly some *global* property concerning, for instance, the sizes or the connectivity of some subsets of vertices. One could therefore try to cover a broad variety of these problems under a same umbrella, and hence develop efficient algorithms to solve them at once. With this objective

[1] Corresponding author.

in mind, several definitions of subsets of partitioning problems, where each vertex has to satisfy a local property, as well as extensions of these sets of problems including some global property, have been proposed and shown to be solvable in polynomial time in various graph classes. In particular, in [6], the authors defined so-called *r-locally checkable problems*. Each of these problems has an associated set of colors and a *check function*, that is, a function that takes as input a vertex v of the graph and a coloring of the r-neighborhood of v (i.e. the set of vertices at distance at most r from v) and outputs True or False. A *proper coloring* of the input graph G is defined as a coloring c of the vertices such that, for every vertex v, the check function applied to v and the restriction of c to the r-neighborhood of v outputs True. They also consider a set of weights with a total order, and associate a weight to each pair of vertex and possible color. The weight of a coloring c is then naturally obtained by combining the weights of the pairs $(v, c(v))$. Then, an r-locally checkable problem consists in finding the minimum weight of a proper coloring of the input graph G. Examples of r-locally checkable problems include k-Coloring, Maximum Independent Set and Minimum Dominating Set [6].

Since many r-locally checkable problems are hard on general graphs, it is of interest to determine under which conditions (on the check function and the set of colors) we can efficiently solve them for a given class of graphs. In [6], the authors showed that, under mild conditions, r-locally checkable problems can be solved in polynomial time in graphs of bounded tree-width. In this paper, we will focus on 1-locally checkable problems with an associated *color-counting* check function, defined as follows.

▶ **Definition 1.** *Let G be a graph and $\textsc{Colors} = \{a_1, \ldots, a_q\}$ be a set of colors. A check function f is* color-counting *if it only depends on the vertex v, the color it receives and, for each color $a \in \textsc{Colors}$, the number of neighbors of v of color a.*

In other words, a check function f is color-counting *if there exists a function f' such that*

$$f(v, c) = f'(v, c(v), n_1, \ldots, n_q)$$

for every vertex $v \in V(G)$ and every coloring c of the closed neighborhood of v, where $n_j = |\{u \in N_G(v) : c(u) = a_j\}|$ for all $j \in \{1, \ldots, q\}$.

Since we are only going to work with color-counting check functions in this paper, we will directly refer to them as $check(v, a, n_1, \ldots, n_q)$.

In [17], the authors analyzed the restrictions on 1-locally checkable problems with respect to mim-width. They define *d-stable* check functions, which are a subset of the color-counting check functions, and we will use them to improve our complexity results.

▶ **Definition 2** ([17])**.** *Let $d \in \mathbb{N}$. Let G be a graph, $\textsc{Colors}$ be a set of q colors and check be a color-counting check function. We say that check is* d-stable *if for all $v \in V(G)$, $a \in \textsc{Colors}$ and non-negative integers $n_1, \ldots, n_q$ we have*

$$check(v, a, n_1, \ldots, n_q) = check(v, a, \min(d, n_1), \ldots, \min(d, n_q)).$$

We present a dynamic programming algorithm, which is XP parameterized by clique-width, for 1-locally checkable problems with a constant number of colors and a color-counting check function. Moreover, this algorithm is FPT when the check function is also d-stable, for any constant d. In a second step, we extend our framework in such a way that we are able to ensure that the size of a given color class belongs to some predefined set of integers. By including this global property for as many colors classes as necessary, the application of our

framework allows to obtain first XP algorithms parameterized by clique-width for problems such as k-COMMUNITY, MAX PDS and (GLOBAL) $[k]$-ROMAN DOMINATION, as well as some variants of them. A generalization of this framework to r-locally checkable problems, for any fixed r, would be quite natural, and the authors of this paper are currently working on it.

The set of locally checkable problems considered here is a subset of the one considered in [6], but notice that our assumptions above are not too restrictive. Indeed, if we do not impose these assumptions then, as explained in [6], one obtains locally checkable problems that are NP-hard on complete graphs (which have clique-width at most 2) and thus, it is unlikely to find XP algorithms parameterized by clique-width for this more general class of locally checkable problems.

As mentioned above, several definitions of subsets of partitioning problems have been defined in the literature and shown to be solvable in polynomial time in various graph classes. We cite here some of the corresponding publications that are the most closely related to our work.

In [7, 20, 27], the authors studied a large class of vertex partitioning problems called *locally checkable vertex partitioning* (LCVP) problems. In these problems, a $q \times q$ matrix D is given, where each entry is a finite or cofinite set of integers. A partition of the set of vertices $V_1, \ldots, V_q$ is sought, such that for each $i, j \in \{1, \ldots, q\}$, we have $|N_G(v) \cap V_j| \in D[i, j]$ for all $v \in V_i$. Empty partition classes are allowed. In [27], Telle and Proskurowski solved these problems in polynomial time on graphs of bounded treewidth. This result was generalized in [7], where Bui-Xuan, Telle and Vatshelle gave an algorithm that solves LCVP problems given a decomposition tree of the input graph. In the same paper, they proved that this algorithm is FPT parameterized by boolean-width, and later in [20], Jaffke et al. showed that the same algorithm is XP parameterized by mim-width, when a suitable decomposition tree is given. As shown in [17], every LCVP problem can be modeled as a 1-locally checkable problem with a d-stable check function (where d is as defined in [7]):

$$check(v, a, n_1, \ldots, n_q) = (\forall j \in \{1, \ldots, q\},\, n_j \in D[a, j])\,.$$

While many locally checkable problems are expressible as LCVP problems, there are still some relevant problems that do not admit such a characterization, but do belong to the set of problems we analyze in this paper. Examples include $[k]-$ROMAN DOMINATION and BALANCED k-COMMUNITY, see Section 6.

In [16], Gerber and Kobler studied a variation of LCVP, with two modifications. On one hand they restrict the entries of D to sets of consecutive integers, and on the other hand, they associate to each vertex v a set $\rho(v) \subseteq \{1, \ldots, q\}$ such that $v \in V_i \Rightarrow i \in \rho(v)$. They show that the problems in this framework are XP when parameterized by clique-width. Notice that these problems are also covered by our framework.

In [10], Courcelle, Makowsky and Rotics presented an algorithm which, given as input a graph with an associated clique-width expression, solves problems expressible in a certain variation of Monadic Second-Order Logic, called MSO_1. On graphs of clique-width at most k, the running time of their algorithm is linear in the size of the input graph. However, as pointed out in [14], the multiplicative constant grows extremely fast with k.

Following a similar research line, in their recent article [5], Bergougnoux, Dreier and Jaffke defined an extension of existential MSO_1, which they call *distance neighborhood logic with acyclicity and connectivity constraints* (A&C DN logic). They provided an algorithm that solves problems expressible in this logic, given a suitable branch decomposition of the input graph. The complexity of the algorithm is expressed in terms of the *d-neighborhood* equivalence relation (see [7]), allowing them to state their main result parameterized by

mim-width (XP), tree-width, rank-width or clique-width (FPT), with a single-exponential dependence. As shown in [17], all locally checkable problems with constant number of colors and d-stable check functions, for some constant d, can be expressed in A&C DN logic. However, locally checkable problems with a color-counting check function that is not d-stable for any constant d, and possibly extended with global properties, such as BALANCED k-COMMUNITY, cannot be directly expressed by an A&C DN logic formula of fixed length.

This paper is structured as follows. In Section 2, we give some definitions and notations. In Section 3, we formally present our framework, while in Section 4, we describe the dynamic programming algorithm, prove its correctness and analyse its complexity. Section 5 deals with the extension of our results of the previous section to include the global size property. Finally, in Section 6, we apply our results to some selected problems.

2 Preliminaries

2.1 Algebraic definitions

Let $f\colon D \to C$ be a function and let $S \subseteq D$. We denote by $f|_S$ the function f restricted to the domain S, that is, the function $f|_S\colon S \to C$ is defined as $f|_S(x) = f(x)$ for all $x \in S$. Let D' be a set such that $D \cap D' = \emptyset$, and let $g\colon D' \to C$. We denote by $f \cup g$ the function $h\colon D \cup D' \to C$ such that $h(x) = f(x)$ if $x \in D$, and $h(x) = g(x)$ if $x \in D'$. Note that, since D and D' are disjoint, $f \cup g$ is well defined.

We denote by $[\![a, b]\!]$, with $a, b \in \mathbb{Z}$ and $a \leq b$, the set of all integer numbers greater than or equal to a and less than or equal to b, that is $\{a, a+1, \ldots, b\}$. Furthermore, we use BOOL to denote the set $\{\text{TRUE}, \text{FALSE}\}$.

2.2 Graph theoretical definitions

Throughout this paper, we consider simple, finite and undirected graphs. For graph theoretical notions not defined here, the reader is referred to [28].

The notion of clique-width of a graph G, denoted by $cw(G)$, was first introduced in [11]. It is defined as the minimum number of labels needed to construct G using the following 4 operations:

- creation of a new vertex v with label i (denoted by $i(v)$);
- disjoint union of two labeled graphs G_1 and G_2 (denoted by $G_1 \oplus G_2$);
- join between two labels i and j, $i \neq j$, i.e. adding an edge between every vertex with label i and every vertex with label j (denoted by $\eta_{i,j}$);
- renaming of label i to label j, i.e. every vertex with label i gets label j (denoted by $\rho_{i \to j}$).

Given a graph class $\mathcal{G}$, the clique-width of $\mathcal{G}$ is $cw(\mathcal{G}) = \sup\{cw(G) \mid G \in \mathcal{G}\}$. We say that $\mathcal{G}$ is *of bounded clique-width* if $cw(\mathcal{G}) < \infty$.

A *clique-width expression* is simply a well-formed expression of operations each corresponding to one of the four operations mentioned above. For a clique-width expression e, we denote by G_e the graph constructed by e. If the number of distinct labels used in a clique-width expression e is at most k, then we say it is a *clique-width k-expression*. It was shown in [12] that any graph G admitting a clique-width k-expression also admits an *irredundant clique-width k-expression*, i.e., such that whenever we execute a join operation $\eta_{i,j}$, there are no already existing edges between vertices with label i and vertices with label j.

Consider a clique-width expression e and the corresponding graph G_e. An *expression tree* of G_e is a rooted binary tree T_e defined as follows:

- The nodes of T_e are of four types corresponding to operations $i(\cdot)$, $\oplus$, η and ρ.
- The leaves of T_e correspond to the creation operation $i(\cdot)$.
- A disjoint union node $\oplus$ corresponds to the disjoint union of the graphs associated with its two children.
- A join node $\eta_{i,j}$ corresponds to the graph associated with its unique child in which we make all vertices of label i adjacent to all vertices of label j.
- A relabeling node $\rho_{i\to j}$ corresponds to the graph associated with its unique child in which we change label i to label j.
- The graph G_e corresponds to the graph associated with the root of T_e.

Notice that for every node $t \in V(T_e)$, the subtree of T_e rooted at t defines a clique-width expression e_t the corresponding graph of which, denoted by G_{e_t}, is a subgraph of G_e. We say that e' is a *subexpression* of e if e' is the expression determined by the subtree of T_e rooted at some node $t \in V(T_e)$. Consider any vertex v in G_{e_t} for some $t \in V(T_e)$. Then all neighbors of v in G_e which are not yet neighbors of v in G_{e_t}, i.e. the edges between v and these vertices are only defined by the ancestor operations of t in T_e, are said to be *upcoming neighbors of v with respect to e_t*. Notice that for any two vertices in G_{e_t} having the same label, their sets of upcoming neighbors with respect to e_t are identical.

Let e be a clique-width k-expression, G_e be its corresponding graph and let T_e be an expression tree of G_e. We define the function $\ell_e : V(G) \to [\![1, k]\!]$ such that $\ell_e(v)$ is the final label of v, i.e. the label of v after the operation corresponding to the root of T_e. We also define $\overline{\ell(e)}$ as the set of labels i such that there exists no $v \in V(G_e)$ such that $\ell_e(v) = i$.

In the remaining of our paper, we will only consider irredundant clique-width k-expressions where in any relabeling operation $\rho_{i\to j}(e)$ we have $j \notin \overline{\ell(e)}$. Notice that under these assumptions the total number of operations in such a clique-width expression of a graph G is in $O(|V(G)| + |E(G)|)$.

2.3 Finite-state automata

A *deterministic finite-state automaton* is a five-tuple $(Q, \Sigma, \delta, q_0, F)$ that consists of

- Q: a finite set of *states*,
- Σ: a finite set of *input symbols* (often called the *alphabet*),
- $\delta\colon Q \times \Sigma \to Q$: a *transition function*,
- $q_0 \in Q$: an *initial* (or *start*) *state*, and
- $F \subseteq Q$: a set of *final* (or *accepting*) *states*.

We say that an automaton $M = (Q, \Sigma, \delta, q_0, F)$ *accepts* a string $c_1 \dots c_n$, with $n \geq 1$, if and only if $c_i \in \Sigma$ for all $1 \leq i \leq n$ and $\delta(\dots\delta(\delta(q_0, c_1), c_2)\dots, c_n) \in F$.

For more about automata theory, we refer the reader to [19].

2.4 Weight sets

Let $(\textsc{Weights}, \preceq)$ be a totally ordered set with a maximum element (called $\textsc{Error}$), together with the minimum operation of the order $\preceq$ (called min) and a closed binary operation on $\textsc{Weights}$ (called $\circledast$) that is commutative and associative, has a neutral element and an absorbing element that is equal to $\textsc{Error}$, and the following property is satisfied: $s_1 \preceq s_2 \Rightarrow s_1 \circledast s_3 \preceq s_2 \circledast s_3$ for all $s_1, s_2, s_3 \in \textsc{Weights}$. In such a case, we say that $(\textsc{Weights}, \preceq, \circledast)$ is a *weight set*.

A classic example of a weight set is $(\mathbb{N} \cup \{+\infty\}, \leq, +)$. Notice that the maximum element is $+\infty$ in this case. We could also consider the reversed order of natural weights: $(\mathbb{N} \cup \{-\infty\}, \geq, +)$, where the maximum element is now $-\infty$. Another simple example worth mentioning is $(\{0, 1\}, \leq, \max)$.

3 Color-counting 1-locally checkable problems

Suppose we are given:

- a simple undirected graph G,
- a set $\textsc{Colors} = \{a_1, \ldots, a_q\}$,
- for every $v \in V(G)$, a nonempty set $L_v \subseteq \textsc{Colors}$ of possible colors for v,
- a weight set $(\textsc{Weights}, \preceq, \circledast)$,
- for every $v \in V(G)$ and for every $a \in L_v$, a weight $\mathrm{w}_{v,a} \in \textsc{Weights} - \{\textsc{Error}\}$ of assigning color a to vertex v, and
- a color-counting check function *check*.

We say that a coloring $c : V(G) \to \textsc{Colors}$ is *valid* if $c(v) \in L_v$ for all $v \in V(G)$. The weight of a valid coloring c is $\mathrm{w}(c) = \circledast_{v \in V} \mathrm{w}_{v,c(v)}$. Furthermore, we say that c is a *proper* coloring of G if it is a valid coloring of G and $check(v, c(v), n_1, \ldots, n_q)$ is true for every $v \in V(G)$, where $n_j = |\{u \in N_G(v) : c(u) = a_j\}|$ for all $j \in [\![1, q]\!]$.

A *color-counting 1-locally checkable problem* consists in finding the minimum weight of a proper coloring of the input graph G.

4 Algorithm

Consider a color-counting 1-locally checkable problem Π and let G be the input graph and e_G a clique-width k-expression of G. Let $\mathcal{N} \in [\![1, |V(G)|]\!]$ be an integer such that $check(v, a, n_1, \ldots, n_q) = check(v, a, \min(\mathcal{N}, n_1), \ldots, \min(\mathcal{N}, n_q))$ for all $v \in V(G)$, $a \in \textsc{Colors}$ and non-negative integers $n_1, \ldots, n_q$.

In this section, we present an algorithm which computes the minimum weight of a proper coloring of G by using the expression e_G as well as the notion of (C, N)-coloring defined hereafter.

▶ **Definition 3** ((C, N)-coloring). *Let e be a subexpression of e_G, and let C and N be two matrices in $[\![0, \mathcal{N}]\!]^{k \times q}$. A valid coloring c of G_e is called a (C, N)-coloring of G_e if the following two conditions hold:*

(C1) $\min(\mathcal{N}, |\{v \in V(G_e) : c(v) = a \wedge \ell_e(v) = i\}|) = C[i, a]$ *for all* $i \in [\![1, k]\!]$ *and all* $a \in \textsc{Colors}$;

(C2) *for every vertex* $v \in V(G_e)$ *we have* $check(v, c(v), n_1, \ldots, n_q) = \textsc{True}$, *where* $n_j = \min(\mathcal{N}, N[\ell_e(v), a_j] + |\{u \in N_{G_e}(v) : c(u) = a_j\}|)$ *for every* $j \in [\![1, q]\!]$.

The minimum weight among all possible (C, N)-colorings of G_e is denoted by $\lambda(e, C, N)$, i.e. $\lambda(e, C, N) = \min\{\mathrm{w}(c) : c \text{ is a } (C, N)\text{-coloring of } G_e\}$. Notice that if no such coloring exists then $\lambda(e, C, N) = \textsc{Error}$.

The following lemma explains the link between proper colorings and (C, N)-colorings.

▶ **Lemma 4.** *Let Π be a color-counting 1-locally checkable problem with input graph G and let e_G be a clique-width k-expression of G. Then the minimum weight of a proper coloring of G equals the minimum among all $\lambda(e_G, C, N_0)$, where $N_0 \in [\![0, \mathcal{N}]\!]^{k \times q}$ is the matrix whose elements are all 0 and $C \in [\![0, \mathcal{N}]\!]^{k \times q}$ is any matrix such that $C[i, a] = 0$ for every $i \in \overline{\ell(e_G)}$ and every $a \in \textsc{Colors}$.*

Proof. We will show that for every proper coloring c of G there exists a matrix $C \in [\![0, \mathcal{N}]\!]^{k \times q}$ such that $C[i, a] = 0$ for every $i \in \overline{\ell(e_G)}$ and every $a \in \textsc{Colors}$, and such that $\mathrm{w}(c) \geq \lambda(e_G, C, N_0)$. On the other hand, we will then show that for every matrix $C \in [\![0, \mathcal{N}]\!]^{k \times q}$ such that $C[i, a] = 0$ for every $i \in \overline{\ell(e_G)}$ and every $a \in \textsc{Colors}$, and such that $\lambda(e_G, C, N_0) \neq \textsc{Error}$, there exists a proper coloring c of G such that $\mathrm{w}(c) = \lambda(e_G, C, N_0)$.

Suppose we have a proper coloring c of G. Let $C \in [\![0, \mathcal{N}]\!]^{k \times q}$ be the matrix such that $C[i,a] = \min(\mathcal{N}, |\{v \in V(G) : c(v) = a \wedge \ell_e(v) = i\}|)$ for all $i \in [\![1,k]\!]$ and all $a \in \textsc{Colors}$. Clearly, $C[i,a] = 0$ for every $i \in \overline{\ell(e_G)}$ and every $a \in \textsc{Colors}$. Also, for every $v \in V(G)$ we have that $\mathit{check}(v, c(v), n_1, \ldots, n_q)$ is true, where $n_j = \min(\mathcal{N}, N_0[\ell_e(v), a_j] + |\{u \in N_G(v) : c(u) = a_j\}|)$ for every $j \in [\![1,q]\!]$, because $N_0[\ell_e(v), a_j] = 0$ for every $j \in [\![1,q]\!]$ and c is a proper coloring of G. Therefore, c is a (C, N_0)-coloring of G and so $\mathrm{w}(c) \geq \lambda(e_G, C, N_0)$.

Now suppose we have a matrix $C \in [\![0, \mathcal{N}]\!]^{k \times q}$ such that $C[i,a] = 0$ for every $i \in \overline{\ell(e_G)}$ and every $a \in \textsc{Colors}$, and such that $\lambda(e_G, C, N_0) \neq \textsc{Error}$. Let c be a (C, N_0)-coloring of G of minimum weight (notice that at least one such c exists, since $\lambda(e_G, C, N_0) \neq \textsc{Error}$). We will prove that c is a proper coloring of G. By definition of a (C, N_0)-coloring, c is a valid coloring, so it only remains to prove that $\mathit{check}(v, c|_{N_G[v]})$ is true for every $v \in V(G)$. We know that for every $v \in V(G)$, we have that $\mathit{check}(v, c(v), n_1, \ldots, n_q)$ is true, where $n_j = \min(\mathcal{N}, N_0[\ell_e(v), a_j] + |\{u \in N_G(v) : c(u) = a_j\}|)$ for every $j \in [\![1,q]\!]$. Since $N_0[\ell_e(v), a_j] = 0$ for every $v \in V(G)$ and $j \in [\![1,q]\!]$, we have that $\mathit{check}(v, c|_{N_G[v]}) = \mathit{check}(v, c(v), n_1, \ldots, n_q) = \textsc{True}$, where $n_j = \min(\mathcal{N}, |\{u \in N_G(v) : c(u) = a_j\}|)$ for all $j \in [\![1,q]\!]$. ◀

So Lemma 4 tells us that in order to solve a color-counting 1-locally checkable problem Π, i.e. in order to find a minimum weight of a proper coloring, it is sufficient to find the minimum weight among all (C, N_0)-colorings of the input graph G, where C and N_0 are as described above. Our algorithm is based exactly on this idea, i.e. it determines the minimum among all $\lambda(e_G, C, N_0)$. This is achieved by traversing the binary rooted tree T_{e_G} in a bottom-up fashion and determining in a recursive way the values $\lambda(e, C, N)$, where e is a subexpression of e_G and $C, N \in [\![0, \mathcal{N}]\!]^{k \times q}$. Throughout this recursion, the matrices C and N will intuitively behave in the following way: if we have a proper coloring c of G such that $c|_{V(G_e)}$ is a (C, N)-coloring of G_e, then

- $C[i,a]$ represents the minimum between $\mathcal{N}$ and the number of vertices v in G_e such that $\ell_e(v) = i$ and $c(v) = a$, and
- $N[i,a]$ represents the minimum between $\mathcal{N}$ and the number of vertices $u \in V(G)$ with $c(u) = a$ that are upcoming neighbors with respect to e of every vertex v with $\ell_e(v) = i$.

For the next four lemmas, we will assume that we are given matrices C and N in $[\![0, \mathcal{N}]\!]^{k \times q}$. We will describe the recursive computation of $\lambda(e, C, N)$ by distinguishing four cases depending on the kind of clique-width operation at the root of the tree T_e.

▶ **Lemma 5** (Creating new vertex: $i(v)$)**.** *If there exists $a \in L_v$ such that $C[i,a] = 1$ and $C[j,b] = 0$ for all the other entries $[j,b]$ in C, and if $\mathit{check}(v, a, N[i,a_1], \ldots, N[i,a_q])$ is true, then $\lambda(i(v), C, N) = \mathrm{w}_{v,a}$. Otherwise, $\lambda(i(v), C, N) = \textsc{Error}$.*

Proof. First notice that $G_{i(v)}$ is the graph consisting of a single vertex v with label i. Therefore, if C and N have the above properties (i.e. there exists $a \in L_v$ such that $C[i,a] = 1$ and $C[j,b] = 0$ for all the other entries $[j,b]$ in C, and $\mathit{check}(v, a, N[i,a_1], \ldots, N[i,a_q])$ is true), there is exactly one (C, N)-coloring c of $G_{i(v)}$, defined by $c(v) = a$. Indeed, since $a \in L_v$, it follows that c is a valid coloring. Moreover, conditions (C1) and (C2) are trivially satisfied. Then, $\lambda(i(v), C, N) = \mathrm{w}(c) = \mathrm{w}_{v,a}$.

On the other hand, if C does not have exactly one nonzero entry, or if it is not in row i, or if it is in a column $a \notin L_v$, or if this entry is not equal to 1, then no valid coloring satisfying condition (C1) exists. If for this unique possible choice of color a, $\mathit{check}(v, a, N[i,a_1], \ldots, N[i,a_q])$ is false, then no (C, N)-coloring of $G_{i(v)}$ exists either. Therefore $\lambda(i(v), C, N) = \textsc{Error}$. ◀

▶ **Lemma 6** (Disjoint union: $e_1 \oplus e_2$). *Let N_1 and N_2 be two matrices in $[\![0,\mathcal{N}]\!]^{k\times q}$ such that:*

- *$N_1[i,a] = 0$ for every label $i \in \overline{\ell(e_1)}$ and every color $a \in$ Colors;*
- *$N_1[i,a] = N[i,a]$ for every label $i \notin \overline{\ell(e_1)}$ and every color $a \in$ Colors; and*
- *N_2 is defined analogously with respect to e_2.*

Then

$$\begin{aligned}\lambda(e_1 \oplus e_2, C, N) = \min\{&\lambda(e_1, C_1, N_1) \circledast \lambda(e_2, C_2, N_2) :\\ &\text{(a) } C_1, C_2 \in [\![0,\mathcal{N}]\!]^{k\times q}\\ &\text{(b) } C_1[i,a] = 0 \text{ for all } i \in \overline{\ell(e_1)}, a \in \textsc{Colors};\\ &\text{(c) } C_2[i,a] = 0 \text{ for all } i \in \overline{\ell(e_2)}, a \in \textsc{Colors};\\ &\text{(d) } C[i,a] = \min(\mathcal{N}, C_1[i,a] + C_2[i,a]) \text{ for all } i \in [\![1,k]\!], a \in \textsc{Colors}\}.\end{aligned}$$

Proof. Let $\alpha = \min\{\lambda(e_1, C_1, N_1) \circledast \lambda(e_2, C_2, N_2) : (a), (b), (c), (d) \text{ are satisfied}\}$. We will first prove that $\lambda(e_1 \oplus e_2, C, N) \geq \alpha$. If $\lambda(e_1 \oplus e_2, C, N) = \textsc{Error}$, then we are done. So assume that $\lambda(e_1 \oplus e_2, C, N) \neq \textsc{Error}$ and let c be a (C,N)-coloring of $G_{e_1\oplus e_2}$ whose weight equals $\lambda(e_1 \oplus e_2, C, N)$. We need to show that there exist C_1 and C_2 in $[\![0,\mathcal{N}]\!]^{k\times q}$ satisfying $(b), (c), (d)$ and such that the weight of c is at least $\lambda(e_1, C_1, N_1) \circledast \lambda(e_2, C_2, N_2)$. Let $c_1 = c|_{V(G_{e_1})}$ and $c_2 = c|_{V(G_{e_2})}$. Then, we define $C_1[i,a] = \min(\mathcal{N}, |\{v \in V(G_{e_1}) : c_1(v) = a \wedge \ell_{e_1}(v) = i\}|)$ for any label $i \in [\![1,k]\!]$ and color $a \in \textsc{Colors}$, and similarly for C_2 with respect to e_2. Consequently, conditions (b) and (c) are satisfied: if $i \in \overline{\ell(e_1)}$, then $\{v \in V(G_{e_1}) : \ell_{e_1}(v) = i\} = \emptyset$, thus $C_1[i,a] = 0$, and similarly for C_2. Condition (d) is also satisfied because c is a (C,N)-coloring of $G_{e_1\oplus e_2}$, $V(G_{e_1})$ and $V(G_{e_2})$ are disjoint, $\ell_{e_1}(v) = \ell_{e_1\oplus e_2}(v)$ for every $v \in V(G_{e_1})$ and $\ell_{e_2}(v) = \ell_{e_1\oplus e_2}(v)$ for every $v \in V(G_{e_2})$. We now show that c_1 is a (C_1, N_1)-coloring of G_{e_1}. Condition (C1) is trivially satisfied by the definition of C_1. To show that condition (C2) is satisfied, we will show that $\mathit{check}(v, c_1(v), n'_1, \ldots, n'_q)$ is true for every vertex $v \in V(G_{e_1})$, where $n'_j = \min(\mathcal{N}, N_1[\ell_{e_1}(v), a_j] + |\{u \in N_{G_{e_1}}(v) : c_1(u) = a_j\}|)$ for every $j \in [\![1,q]\!]$. Since c is a (C,N)-coloring of $G_{e_1\oplus e_2}$, we have that $\mathit{check}(v, c(v), n_1, \ldots, n_q)$ is true for every $v \in V(G_{e_1\oplus e_2})$, where $n_j = \min(\mathcal{N}, N[\ell_{e_1\oplus e_2}(v), a_j] + |\{u \in N_{G_{e_1\oplus e_2}}(v) : c(u) = a_j\}|)$ for every $j \in [\![1,q]\!]$. By definition of N_1 and since $\ell_{e_1}(v) = \ell_{e_1\oplus e_2}(v)$ for every $v \in V(G_{e_1})$, we have $N_1[\ell_{e_1}(v), a_j] = N[\ell_{e_1\oplus e_2}(v), a_j]$ for every $v \in V(G_{e_1})$ and every $j \in [\![1,q]\!]$. Also, $N_{G_{e_1}}(v) = N_{G_{e_1\oplus e_2}}(v)$ for every $v \in V(G_{e_1})$ and $c_1 = c|_{V(G_{e_1})}$, so $|\{u \in N_{G_{e_1}}(v) : c_1(u) = a_j\}| = |\{u \in N_{G_{e_1\oplus e_2}}(v) : c(u) = a_j\}|$ for every $v \in V(G_{e_1})$ and every $j \in [\![1,q]\!]$. Therefore $n'_j = n_j$ for every $j \in [\![1,q]\!]$ and hence, $\mathit{check}(v, c_1(v), n'_1, \ldots, n'_q)$ is true for every $v \in V(G_{e_1})$. Using similar arguments, we can show that c_2 is a (C_2, N_2)-coloring of G_{e_2}. Moreover, $\mathrm{w}(c) = \mathrm{w}(c_1) \circledast \mathrm{w}(c_2)$ by definition of c_1 and c_2, and consequently, $\lambda(e_1 \oplus e_2, C, N) = \mathrm{w}(c) = \mathrm{w}(c_1) \circledast \mathrm{w}(c_2) \geq \lambda(e_1, C_1, N_1) \circledast \lambda(e_2, C_2, N_2) \geq \alpha$.

Let us show now that $\lambda(e_1 \oplus e_2, C, N) \leq \alpha$. Consider two matrices C_1 and C_2 in $[\![0,\mathcal{N}]\!]^{k\times q}$ satisfying $(b), (c), (d)$. If $\lambda(e_1, C_1, N_1) = \textsc{Error}$ or $\lambda(e_2, C_2, N_2) = \textsc{Error}$, then we are done. Otherwise, we are going to construct a (C,N)-coloring c of $G_{e_1\oplus e_2}$ such that $\mathrm{w}(c) = \lambda(e_1, C_1, N_1) \circledast \lambda(e_2, C_2, N_2)$. Let c_1 be a (C_1, N_1)-coloring of G_{e_1} whose weight is $\lambda(e_1, C_1, N_1)$ and c_2 be a (C_2, N_2)-coloring of G_{e_2} whose weight is $\lambda(e_2, C_2, N_2)$. Let $c = c_1 \cup c_2$ (note that c is well defined because $V(G_{e_1})$ and $V(G_{e_2})$ are disjoint sets). Clearly, c is a valid coloring and $\mathrm{w}(c) = \mathrm{w}(c_1) \circledast \mathrm{w}(c_2)$. We now show that c is a (C,N)-coloring of $G_{e_1\oplus e_2}$. We start with condition (C1). Consider $i \in [\![1,k]\!]$ and $a \in \textsc{Colors}$. Since c_1 is a (C_1, N_1)-coloring of G_{e_1} and c_2 is a (C_2, N_2)-coloring of G_{e_2}, we have $C_1[i,a] = \min(\mathcal{N}, |\{v \in V(G_{e_1}) : c_1(v) = a \wedge \ell_{e_1}(v) = i\}|)$ and $C_2[i,a] = \min(\mathcal{N}, |\{v \in V(G_{e_2}) : c_2(v) = a \wedge \ell_{e_2}(v) = i\}|)$.

Then,

$$\begin{aligned}
C[i,a] &= \min(\mathcal{N}, C_1[i,a] + C_2[i,a]) \\
&= \min(\mathcal{N}, \min(\mathcal{N}, |\{v \in V(G_{e_1}) : c_1(v) = a \wedge \ell_{e_1}(v) = i\}|) + \\
&\quad \min(\mathcal{N}, |\{v \in V(G_{e_2}) : c_2(v) = a \wedge \ell_{e_2}(v) = i\}|)) \\
&= \min(\mathcal{N}, |\{v \in V(G_{e_1}) : c_1(v) = a \wedge \ell_{e_1}(v) = i\}| + \\
&\quad |\{v \in V(G_{e_2}) : c_2(v) = a \wedge \ell_{e_2}(v) = i\}|) \\
&= \min(\mathcal{N}, |\{v \in V(G_{e_1 \oplus e_2}) : c(v) = a \wedge \ell_{e_1 \oplus e_2}(v) = i\}|).
\end{aligned}$$

The last equality is simply due to the definition of c and to the facts that $V(G_{e_1 \oplus e_2}) = V(G_{e_1}) \cup V(G_{e_2})$ and for every $v \in G_{e_1}$ we have that $\ell_{e_1}(v) = \ell_{e_1 \oplus e_2}(v)$ and for every $v \in G_{e_2}$ we have that $\ell_{e_2}(v) = \ell_{e_1 \oplus e_2}(v)$. Let us focus on (C2) now. Consider $v \in G_{e_1 \oplus e_2}$. Assume, without loss of generality, that $v \in V(G_{e_1})$. We will prove that $check(v, c(v), n_1, \ldots, n_q)$ is true, where $n_j = \min(\mathcal{N}, N[\ell_{e_1 \oplus e_2}(v), a_j] + |\{u \in N_{G_{e_1 \oplus e_2}}(v) : c(u) = a_j\}|)$ for every $j \in [\![1, q]\!]$. Since c_1 is a (C_1, N_1)-coloring of G_{e_1}, we know that $check(v, c_1(v), n'_1, \ldots, n'_q)$ is true, where $n'_j = \min(\mathcal{N}, N_1[\ell_{e_1}(v), a_j] + |\{u \in N_{G_{e_1}}(v) : c_1(u) = a_j\}|)$ for every $j \in [\![1, q]\!]$. Using similar arguments as above, we obtain again that $n'_j = n_j$ for every $j \in [\![1, q]\!]$. Due to the definition of c, we then conclude that $check(v, c(v), n_1, \ldots, n_q)$ is true for every $v \in V(G)$, and so (C2) is satisfied. Thus, c is a (C, N)-coloring of $G_{e_1 \oplus e_2}$. ◀

▶ **Lemma 7** (Join: $\eta_{i,j}(e)$)**.** *Let $N_e \in [\![0, \mathcal{N}]\!]^{k \times q}$ be such that*

- $N_e[i,a] = \min(\mathcal{N}, N[i,a] + C[j,a])$ *for every* $a \in$ Colors*;*
- $N_e[j,a] = \min(\mathcal{N}, N[j,a] + C[i,a])$ *for every* $a \in$ Colors*;*
- $N_e[h,a] = N[h,a]$ *for every* $h \in [\![1,k]\!] \setminus \{i,j\}$ *and every* $a \in$ Colors*.*

Then, $\lambda(\eta_{i,j}(e), C, N) = \lambda(e, C, N_e)$.

Proof. First of all, since the labelings of the vertices do not change between $G_{\eta_{i,j}(e)}$ and G_e, we simply use ℓ to denote both labelings ℓ_e and $\ell_{\eta_{i,j}(e)}$. Also, since $V(G_{\eta_{i,j}(e)}) = V(G_e)$, we simply denote this set by V.

Let $c\colon V \to$ Colors be a valid coloring. We are going to show that c is a (C, N)-coloring of $G_{\eta_{i,j}(e)}$ if and only if c is a (C, N_e)-coloring of G_e. In order to prove this, it suffices to show that, for every color $a \in$ Colors and every vertex $v \in V$, we have $\min(\mathcal{N}, N[\ell(v), a] + |\{u \in N_{G_{\eta_{i,j}(e)}}(v) : c(u) = a\}|) = \min(\mathcal{N}, N_e[\ell(v), a] + |\{u \in N_{G_e}(v) : c(u) = a\}|)$. If $\ell(v) \neq i, j$ then $N_e[\ell(v), a_i] = N[\ell(v), a_i]$ by definition of N_e and clearly $N_{G_e}(v) = N_{G_{\eta_{i,j}(e)}}(v)$. On the other hand, if $\ell(v) = i$ (if $\ell(v) = j$ the proof is analogous) then

$$\begin{aligned}
&\min(\mathcal{N}, N_e[i,a] + |\{u \in N_{G_e}(v) : c(u) = a\}|) \\
&= \min(\mathcal{N}, \min(\mathcal{N}, N[i,a] + C[j,a]) + |\{u \in N_{G_e}(v) : c(u) = a\}|) \\
&= \min(\mathcal{N}, N[i,a] + C[j,a] + |\{u \in N_{G_e}(v) : c(u) = a\}|) \\
&= \min(\mathcal{N}, N[i,a] + \min(\mathcal{N}, |\{u \in V : c(u) = a \wedge \ell(u) = j\}|) + |\{u \in N_{G_e}(v) : c(u) = a\}|) \\
&= \min(\mathcal{N}, N[i,a] + |\{u \in V : c(u) = a \wedge \ell(u) = j\}| + |\{u \in N_{G_e}(v) : c(u) = a\}|) \\
&= \min(\mathcal{N}, N[i,a] + |\{u \in N_{G_{\eta_{i,j}(e)}}(v) : c(u) = a \wedge \ell(u) = j\}| + |\{u \in N_{G_e}(v) : c(u) = a\}|) \\
&= \min(\mathcal{N}, N[i,a] + |\{u \in N_{G_{\eta_{i,j}(e)}}(v) : c(u) = a\}|).
\end{aligned}$$

The last two equalities follow from the fact that every vertex with label j is a neighbor of v in $G_{\eta_{i,j}(e)}$ but a non-neighbor of v in G_e (recall that we are working with irredundant clique-width expressions). ◀

▶ **Lemma 8** (Relabeling: $\rho_{i\to j}(e)$)**.** *Let N_e be such that*

- $N_e[i,a] = N[j,a]$ *for every* $a \in$ Colors*;*
- $N_e[h,a] = N[h,a]$ *for every* $h \in [\![1,k]\!] \setminus \{i\}$ *and every* $a \in$ Colors*.*

If $C[i,a] = 0$ for all $a \in$ Colors, then

$$\begin{aligned}\lambda(\rho_{i\to j}(e), C, N) = \min\{\lambda(e, C_e, N_e) : \\ &\text{(a) } C_e \in [\![0,\mathcal{N}]\!]^{k\times q} \\ &\text{(b) } C[j,a] = \min(\mathcal{N}, C_e[i,a] + C_e[j,a]) \text{ for all } a \in \text{Colors}; \\ &\text{(c) } C_e[h,a] = C[h,a] \text{ for all } h \in [\![1,k]\!] \setminus \{i,j\}, a \in \text{Colors}\}.\end{aligned}$$

Otherwise, $\lambda(\rho_{i\to j}(e), C, N) =$ Error.

Proof. If $C[i,a] \neq 0$ for some $a \in$ Colors, then, by definition, there exists no (C,N)-coloring of $G_{\rho_{i\to j}(e)}$ and $\lambda(\rho_{i\to j}(e), C, N) =$ Error. So we may assume now that $C[i,a] = 0$ for all $a \in$ Colors. Let $\alpha = \min\{\lambda(e, C_e, N_e) : (a), (b), (c) \text{ are satisfied}\}$. We will first prove that $\lambda(\rho_{i\to j}(e), C, N) \geq \alpha$. Let c be a (C,N)-coloring of $G_{\rho_{i\to j}(e)}$ whose weight equals $\lambda(\rho_{i\to j}(e), C, N)$. We will show that there exists a matrix C_e in $[\![0,\mathcal{N}]\!]^{k\times q}$ satisfying $(b), (c)$ and such that c is a (C_e, N_e)-coloring of G_e. Let

$$C_e[h,a] = \min(\mathcal{N}, |\{v \in V(G_e) : c(v) = a \wedge \ell_e(v) = h\}|)$$

for every $h \in [\![1,k]\!]$ and every $a \in$ Colors. Since c is a (C,N)-coloring of $G_{\rho_{i\to j}(e)}$, then $C[h,a] = \min(\mathcal{N}, |\{v \in V(G_{\rho_{i\to j}(e)}) : c(v) = a \wedge \ell_{\rho_{i\to j}(e)}(v) = h\})$ for every $h \in [\![1,k]\!]$ and $a \in$ Colors. Therefore, (c) is trivially satisfied, since $\ell_{\rho_{i\to j}(e)}(v) = \ell_e(v)$ for any vertex v whose label is neither i nor j in G_e. Furthermore the following inequalities hold,

$$\begin{aligned}\min(\mathcal{N}, C_e[i,a] + C_e[j,a]) =& \min(\mathcal{N}, \min(\mathcal{N}, |\{v \in V(G_e) : c(v) = a \wedge \ell_e(v) = i\}|) \\ &+ \min(\mathcal{N}, |\{v \in V(G_e) : c(v) = a \wedge \ell_e(v) = j\}|)) \\ =& \min(\mathcal{N}, |\{v \in V(G_e) : c(v) = a \wedge \ell_e(v) = i\}| \\ &+ |\{v \in V(G_e) : c(v) = a \wedge \ell_e(v) = j\}|) \\ =& \min(\mathcal{N}, |\{v \in V(G_e) : c(v) = a \wedge (\ell_e(v) = i \vee \ell_e(v) = j)\}|) \\ =& \min(\mathcal{N}, |\{v \in V(G_{\rho_{i\to j}(e)}) : c(v) = a \wedge \ell_{\rho_{i\to j}(e)}(v) = j\}|) \\ =& C[j,a]\end{aligned}$$

and thus, condition (b) is satisfied as well. We next show that c is a (C_e, N_e)-coloring of G_e. We know that c is a valid coloring because c is a (C,N)-coloring of $G_{\rho_{i\to j}(e)}$. Also, (C1) is trivially satisfied from our definition of C_e. For (C2), we have to show that $check(v, c(v), n'_1, \ldots, n'_q)$ is true for every v in G_e, where $n'_b = \min(\mathcal{N}, N_e[\ell_e(v), a_b] + |\{u \in N_{G_e}(v) : c(u) = a_b\}|)$ for every $v \in V(G_e)$ and every $b \in [\![1,q]\!]$. Since c is a (C,N)-coloring of $G_{\rho_{i\to j}(e)}$, we know that $check(v, c(v), n_1, \ldots, n_q)$ is true for every $v \in V(G_{\rho_{i\to j}(e)})$, where $n_b = \min(\mathcal{N}, N[\ell_{\rho_{i\to j}(e)}(v), a_b] + |\{u \in N_{G_{\rho_{i\to j}(e)}}(v) : c(u) = a_b\}|)$ for every $b \in [\![1,q]\!]$. By definition of N_e and since $N_{G_e}(v) = N_{G_{\rho_{i\to j}(e)}}(v)$ for all vertices v, we have $n_b = n'_b$ for all $b \in [\![1,q]\!]$, and therefore $check(v, c(v), n'_1, \ldots, n'_q)$ is true for every $v \in G_e$ and $b \in [\![1,q]\!]$.

We now show that $\lambda(\rho_{i\to j}(e), C, N) \leq \alpha$. Let C_e be a matrix in $[\![0,\mathcal{N}]\!]^{k\times q}$ satisfying $(b), (c)$, and let c be a (C_e, N_e)-coloring of G_e of weight $\lambda(e, C_e, N_e)$. We are going to show that c is also a (C,N)-coloring of $G_{\rho_{i\to j}(e)}$. Condition (C1) is trivially satisfied for every label h in $[\![1,k]\!] \setminus \{i,j\}$ and every color a. For label i, condition (C1) is also true since $C[i,a] = 0$ by assumption and there are no vertices with label i in $V(G_{\rho_{i\to j}(e)})$. We can show in a

Algorithm 1 Main algorithm.

```
1  for every subexpression e of e_G, traversing them in a bottom-up fashion, do
2  |  forall matrices C, N ∈ [[0, 𝒩]]^{k×q} do
3  |  |  Compute λ(e, C, N) using Lemmas 5, 6, 7 or 8, according to the type of the
   |  |   operation at the root of T_e.
4  |  |  Store the result in memory for future uses.
5  |  end
6  end
7  Let N_0 be the matrix in [[0, 𝒩]]^{k×q} such that all its elements are 0.
8  Let m ← Error
9  forall C ∈ [[0, 𝒩]]^{k×q} such that C[i, a] = 0 for every i ∈ ℓ(e_G), a ∈ Colors do
10 |  m ← min(m, λ(e_G, C, N_0))
11 end
12 return m
```

similar way as above that $C[j,a] = \min(\mathcal{N}, |\{v \in V(G_{\rho_{i\to j}(e)}) : c(v) = a \wedge \ell_{\rho_{i\to j}(e)}(v) = j\}|)$ for all $a \in$ Colors. We need to verify now (C2), i.e., for every vertex v, we have that $check(v, c(v), n_1, \ldots, n_q)$ is true, where $n_b = \min(\mathcal{N}, N[\ell_{\rho_{i\to j}(e)}(v), a_b] + |\{u \in N_{G_{\rho_{i\to j}(e)}}(v) : c(u) = a_b\}|)$. As before, this is a consequence of the fact that c is a (C_e, N_e)-coloring of G_e, and that for every vertex v and color a_b, we have $N_e[\ell_e(v), a_b] = N[\ell_{\rho_{i\to j}(e)}(v), a_b]$ by definition and $|\{u \in N_{G_e}(v) : c(u) = a_b\}|) = |\{u \in N_{G_{\rho_{i\to j}(e)}}(v) : c(u) = a_b\}|$. ◀

Our algorithm, which takes the same input as a locally checkable problem, plus the number $\mathcal{N}$ and an irredundant clique-width k-expression e_G of the input graph G together with its binary rooted tree T_{e_G}, and outputs the minimum weight of a proper coloring of G, is presented in Algorithm 1. As explained above, we proceed in a bottom-up fashion, i.e. we start with the leaf nodes of T_{e_G}, then continue with their parents and so on, and compute each time $\lambda(e, C, N)$ for the corresponding subexpression e (i.e. for the subexpression e corresponding to the node of T_{e_G} that we are currently analyzing) and all possible choices of C and N using the recurrences in Lemmas 5, 6, 7 and 8 (see lines 1-3). Since we are storing the results (see line 4), the number of times we need to compute some value $\lambda(\cdot,\cdot,\cdot)$ is given by the number of subexpressions of e_G times the possible choices for the matrices C and N. Since we have $O(|V(G)| + |E(G)|)$ subexpressions in the given clique-width expression (see Section 2), and since there exist $(\mathcal{N}+1)^{kq}$ possible matrices C, respectively possible matrices N, we obtain that line 3 of our algorithm is called at most $O((|V(G)| + |E(G)|)(\mathcal{N}+1)^{2kq})$ times. In lines 7-11, we then determine the minimum among all $\lambda(e_G, C, N_0)$, where N_0 is the matrix whose elements are all 0, and $C \in [\![0, \mathcal{N}]\!]^{k\times q}$ is any matrix such that $C[i,a] = 0$ for every $i \in \overline{\ell(e_G)}$ and every $a \in$ Colors. This can be done in time $O((\mathcal{N}+1)^{kq})$.

It remains to determine the complexity of computing some value $\lambda(\cdot,\cdot,\cdot)$. This clearly depends on the operation we consider. Thus, we distinguish 4 cases:

- **Creating new vertex:** We need to go through the entries of C, which can be done in time $O(kq)$. Let us denote by $t_{check}(|V(G)|, q, \mathcal{N})$ the complexity of evaluating the check function. Hence, we obtain a complexity of $O(kq + t_{check}(|V(G)|, q, \mathcal{N}))$ for this operation.
- **Disjoint union:** We first need to determine N_1 and N_2, which takes $O(kq)$ time, and then we need to find the minimum weight by going through all possible choices of C_1 and C_2, which can be done in time $O((\mathcal{N}+1)^{2kq})$. This gives us an overall complexity of $O((\mathcal{N}+1)^{2kq})$ for determining $\lambda(\cdot,\cdot,\cdot)$ for the disjoint union operation.

- **Join:** We simply need to determine the matrix N_e, which can be done in $O(kq)$ time.
- **Relabeling:** First, we need to determine the matrix N_e, which takes $O(kq)$ time, and then we need to find the minimum weight by considering possible choices of C_e with all rows fixed except two, which clearly takes $O((\mathcal{N}+1)^{2q})$. Thus, overall the complexity of determining $\lambda(\cdot,\cdot,\cdot)$ for the relabeling operation is $O(kq+(\mathcal{N}+1)^{2q})$.

Now the complexity of computing any $\lambda(e,C,N)$ is bounded by the sum of the complexities of the four cases, for which we obtain $O(t_{check}(|V(G)|,q,\mathcal{N})+(\mathcal{N}+1)^{2kq})$. Thus, we obtain the following complexity:

$$O((|V(G)|+|E(G)|)(\mathcal{N}+1)^{2kq}(t_{check}(|V(G)|,q,\mathcal{N})+(\mathcal{N}+1)^{2kq})).$$

▶ Remark 9. We can modify the algorithm in order to also obtain the coloring function as an output. This does not affect the complexity.

Let us highlight the following main consequences of the previous analysis. Notice that, by the results in [23], we do not need a clique-width expression as input.

▶ **Corollary 10.** *Consider a color-counting 1-locally checkable problem* Π *with constant number of colors and a check function computable in polynomial time. Then* Π *is XP parameterized by clique-width.*

▶ **Corollary 11.** *Let* $d\in\mathbb{N}$. *If* Π *is a* d*-stable 1-locally checkable problem where the number of colors is* $O(\log|V(G)|)$ *and the check function can be computed in polynomial time, then* Π *is XP parameterized by clique-width.*

▶ **Corollary 12.** *Let* $d\in\mathbb{N}$. *If* Π *is a* d*-stable 1-locally checkable problem where the number of colors is constant and the check function can be computed in constant time, then* Π *is FPT parameterized by clique-width. Moreover, if an irredundant clique-width* k*-expression is given as input, then it is linear FPT parameterized by* k.

Notice that various well-known graph theoretical problems, such as k-Coloring, Maximum Independent Set, as well as $[k]-$Roman domination (see Section 6), are indeed d-stable 1-locally checkable problems, for some constant d, with constant number of colors.

We would also like to mention that Lemmas 4, 5, 6, 7 and 8 still hold if in this section we replace every occurrence of "$min(\mathcal{N},\cdot)$" by "$\cdot \bmod (\mathcal{N}+1)$". Further, the complexity of Algorithm 1 remains the same. By doing so, we obtain FPT algorithms parameterized by clique-width for problems such as parity domination [15, 18] and finding the minimum size of a non-$z(\bmod k)$ dominating set [8]. The next corollary formally states this observation.

▶ **Corollary 13.** *Let* $m\in\mathbb{N}$. *Consider a color-counting 1-locally checkable problem* Π *where the number of colors is constant, and the check function can be computed in constant time and is such that* $check(v,a,n_1,\ldots,n_q)=check(v,a,n_1 \bmod m,\ldots,n_q \bmod m)$ *for all* $v\in V(G)$, $a\in$ Colors *and non-negative integers* $n_1,\ldots,n_q$. *Then* Π *is FPT parameterized by clique-width. Moreover, if an irredundant clique-width* k*-expression is given as input, then it is linear FPT parameterized by* k.

5 Global size property

In this section, we extend the results of Section 3 by considering color-counting 1-locally checkable problems in which it is also required that the number of vertices that receive a given color $a\in$ Colors belongs to a predefined set σ_a of non-negative integers.

Let $(Q, \{1\}, \delta, q_0, F)$ be a deterministic finite-state automaton which accepts a string of t consecutive 1's if and only if $t \in \sigma_a$. Note that for all finite sets of non-negative integers, there exists such an automaton (for example, let m be the maximum element of the set, then we set $Q = \{s_0, \ldots, s_{m+1}\}$, $q_0 = s_0$, $F = \{s_i : i \in \sigma\}$, $\delta(s_i, 1) = s_{i+1}$ for all $0 \leq i \leq m$ and $\delta(s_{m+1}, 1) = s_{m+1}$). Let us define the notation $\delta^0(s_i) = s_i$ and $\delta^n(s_i) = \delta(\delta^{n-1}(s_i), 1)$ for every state $s_i \in Q$ and positive integer n.

We will now proceed in a similar way as in Section 4 but considering additional parameters. Let us first introduce the relevant notion of $(C, N, p_1, \ldots, p_m)$-*colorings*, which will be defined recursively. This notion can be used to extend the results of the aforementioned section using different global properties. Intuitively, if $C, N \in [\![0, \mathcal{N}]\!]^{k \times q}$ and $p_1, \ldots, p_m$ are parameters such that $(C, N, p_1, \ldots, p_m)$-colorings of G_e are defined, then, for additional parameters $p_{m+1}, \ldots, p_{m+m'}$, we define a $(C, N, p_1, \ldots, p_m, p_{m+1}, \ldots, p_{m+m'})$-*coloring of* G_e as a $(C, N, p_1, \ldots, p_m)$-coloring of G_e such that parameters $p_{m+1}, \ldots, p_{m+m'}$ satisfy some predefined property. In the case of the particular global property mentioned at the beginning of this section, we will only consider two additional parameters. The first such parameter is a state $s_a \in Q$ and the second parameter is a function $f_a \colon Q \to$ Bool. We then define a $(C, N, p_1, \ldots, p_m, s_a, f_a)$-*coloring* c of G_e as a $(C, N, p_1, \ldots, p_m)$-coloring of G_e such that $f_a(\delta^n(s_a)) =$ True, where $n = |\{v \in V(G_e) : c(v) = a\}|$. Also, in the same spirit as before, we will denote by $\lambda(e, C, N, p_1, \ldots, p_m, s_a, f_a)$ the minimum weight among all $(C, N, p_1, \ldots, p_m, s_a, f_a)$-colorings of G_e. If we want to fix the size of $\mathcal{R}$ color classes, say $a_1, \ldots, a_{\mathcal{R}}$, it suffices to associate an automaton M_i and the corresponding parameters s_{a_i} and f_{a_i} with each color class a_i, for $i \in [\![1, \mathcal{R}]\!]$.

By providing a lemma explaining how to solve a color-counting 1-locally checkable problem with given global properties by using $(C, N, p_1, \ldots, p_m, s_a, f_a)$-colorings, and then again distinguishing the four clique-width operations, as in the previous section, we can prove that, when the number of colors is constant, this new algorithm is also XP parameterized by clique-width. Due to space restrictions, their statements are omitted here, but presented in Appendix A.

6 Applications

In this section, we provide some examples of problems whose complexity status in graphs of bounded clique-width was unknown, and for each of which the application of our framework yields a first polynomial-time algorithm in this class of graphs.

6.1 (Global) $[k]$-Roman domination

The $[k]$-Roman domination problem was first defined in [1] as a generalization of Roman and double Roman domination [9, 4]. Let $k \geq 1$ be an integer. A $[k]$-*Roman dominating function* on a graph G is a function $f \colon V(G) \to [\![0, k+1]\!]$ having the property that if $f(v) < k$ then $\sum_{u \in N_G[v]} f(u) \geq |AN_G^f(v)| + k$, where $AN_G^f(v) = \{u \in N_G(v) : f(u) \geq 1\}$ (this set is called the *active neighborhood of* v). The *weight* of a $[k]$-Roman dominating function f is $\sum_{v \in V(G)} f(v)$, and the minimum weight of a $[k]$-Roman dominating function on G is the $[k]$-*Roman domination number* of G, denoted by $\gamma_{[kR]}(G)$. The problem consists in computing the $[k]$-Roman domination number of a given graph.

In [6], this problem was shown to be solvable in linear time in graphs of bounded treewidth. In their model, the number of colors is a constant and the check function is actually $(k+1)$-stable. We can express it in the following way:

- $\textsc{Colors} = [\![0, k+1]\!]$ and $L_v = [\![0, k+1]\!]$ for all $v \in V(G)$;
- $(\textsc{Weights}, \preceq, \circledast) = (\mathbb{N} \cup \{+\infty\}, \leq, +)$ and $\mathrm{w}_{v,a} = a$ for all $v \in V(G), a \in L_v$;
- $check(v, a, n_0, \ldots, n_{k+1}) = \left(a + \sum_{j=0}^{k+1} j n_j \geq k + \sum_{j=1}^{k+1} n_j\right)$.

Then, by Corollary 12, this problem is FPT parameterized by clique-width (and linear FPT when a suitable clique-width expression is given).

In [25], the authors introduced a variant of this problem, called GLOBAL ROMAN DOMINATION. This problem was later extended to GLOBAL DOUBLE ROMAN DOMINATION [26] and GLOBAL TRIPLE ROMAN DOMINATION [21]. The definition of these problems can be naturally generalized as follows. A *global* $[k]$*-Roman dominating function* on a graph G is a $[k]$-Roman dominating function in both G and $\overline{G}$. The GLOBAL $[k]$-ROMAN DOMINATION problem consists in computing the minimum weight of a global $[k]$-Roman dominating function of a graph.

In order to show that this problem is XP parameterized by clique-width, we first define an auxiliary problem.

SPECIFIED SIZE GLOBAL $[k]$-ROMAN DOMINATION
Instance: A graph G and $k+2$ non-negative integers $s_0, \ldots, s_{k+1}$ such that $\sum_{i=0}^{k+1} s_i = |V(G)|$.
Question: Does G admit a global $[k]$-Roman dominating function f such that, for all $i \in [\![0, k+1]\!]$, s_i equals the number of vertices $v \in V(G)$ with $f(v) = i$?

This last problem can be modeled as a color-counting 1-locally checkable problem with global properties:

- $\textsc{Colors} = [\![0, k+1]\!]$ and $L_v = [\![0, k+1]\!]$ for all $v \in V(G)$;
- $(\textsc{Weights}, \preceq, \circledast) = (\mathbb{N} \cup \{+\infty\}, \leq, +)$ and $\mathrm{w}_{v,a} = a$ for all $v \in V(G), a \in L_v$;
- $check(v, a, n_0, \ldots, n_{k+1}) = (a + \sum_{j=1}^{k+1}(j-1)n_j \geq k) \wedge (\sum_{j=1}^{k+1}(j-1)(s_j - n_j) \geq k)$;
- for all $a \in \textsc{Colors}$, we ask for the size of the color class of a to belong to $\{s_a\}$.

Finally, to solve GLOBAL $[k]$-ROMAN DOMINATION on graphs of bounded clique-width, we successively iterate over the feasible combinations of values $s_0, \ldots, s_{k+1}$ such that $\sum_{i=0}^{k+1} s_i = |V(G)|$ and $s_i \geq 0$ for all $i \in [\![0, k+1]\!]$. Notice that the number of such combinations is no more than $(|V(G)|+1)^{k+2}$. For each combination, we solve SPECIFIED SIZE GLOBAL $[k]$-ROMAN DOMINATION, and we retain the solution of minimum weight.

6.2 k-community, Max PDS and other variants

The notion of *community structure* was first introduced in [22], as a partition $\{C_1, \ldots, C_k\}$, with $k \geq 2$, of the set of vertices of a graph into so called *communities*, such that for each $i \in [\![1, k]\!]$ we have $|C_i| \geq 2$ and, for each vertex $v \in C_i$ and each community $C_j \neq C_i$, $\frac{|N_G(v) \cap C_i|}{|C_i|-1} \geq \frac{|N_G(v) \cap C_j|}{|C_j|}$. Finding a community structure in any graph G can be done in polynomial time (see [22]). However, the number of communities k in the obtained community structure can be any value between 2 and $\frac{|V(G)|}{2}$, and the algorithm does not apply when we want to impose the number of communities. The 2-COMMUNITY problem was introduced in [2] as the problem of deciding whether a given connected graph has a *2-community structure*, i.e. a community structure with 2 communities. This can be naturally generalized to the k-COMMUNITY problem, for any fixed k, as the problem of deciding whether a given connected graph has a community structure with k communities. The complexity status of 2-COMMUNITY is still unknown, and only a few graph classes are known to admit polynomial time algorithms for this problem (for instance, graphs of maximum degree 3 and graphs of minimum degree $|V(G)| - 3$ [2]).

We show here that k-COMMUNITY is XP parameterized by clique-width. Our approach is similar to the one for GLOBAL $[k]-$ROMAN DOMINATION, in the sense that we define a variant of the problem where we require a certain size of each community, to which we reduce k-COMMUNITY.

SPECIFIED SIZE k-COMMUNITY
Instance: A graph G and k integers $s_1, \ldots, s_k \geq 2$, such that $\sum_{i=1}^{k} s_i = |V(G)|$.
Question: Does G admit a k-community structure $\{C_1, \ldots, C_k\}$ such that $|C_i| = s_i$ for all $i \in [\![1, k]\!]$?

The SPECIFIED SIZE k-COMMUNITY problem can be modeled as a color-counting 1-locally checkable problem with global properties. Notice that since it is a decision problem, we only need two values for the weight set.

- COLORS $= [\![1, k]\!]$ and $L_v = [\![1, k]\!]$ for all $v \in V(G)$;
- $(\text{WEIGHTS}, \preceq, \circledast) = (\{0, 1\}, \leq, \max)$ and $\text{w}_{v,a} = 0$ for all $v \in V(G), a \in L_v$;
- $check(v, a, n_1, \ldots, n_k) = \left(\forall b \in [\![1, k]\!], \frac{n_a}{s_a - 1} \geq \frac{n_b}{s_b}\right)$;
- for all $a \in$ COLORS, we ask for the size of the color class of a to belong to $\{s_a\}$.

Then, k-COMMUNITY can be solved by successively iterating over the feasible combinations of values $s_1, \ldots, s_k$ such that $\sum_{i=1}^{k} s_i = |V(G)|$ and $s_i \geq 2$ for all $i \in [\![1, k]\!]$, and for each of the combinations solving SPECIFIED SIZE k-COMMUNITY.

Note that BALANCED k-COMMUNITY, i.e. the problem of finding a k-community structure with all parts having the same size, is equivalent to SPECIFIED SIZE k-COMMUNITY with $s_i = s_j$, for all $i, j \in [\![1, k]\!]$. Hence, it is also XP parameterized by clique-width. In [13], it was shown that this problem is NP-complete in general, and in [2] it was pointed out to be polynomially solvable in graphs of bounded treewidth. It is not difficult to see that the problem WEAK k-COMMUNITY, defined in [2], can also be solved by slightly modifying the above check function.

A closely related problem is the MAXIMUM PROPORTIONALLY DENSE SUBGRAPH (MAX PDS) problem, originally defined in [3]. Let G be a graph and $S \subset V(G)$, such that $2 \leq |S| < |V(G)|$. We say that the induced subgraph $G[S]$ is a *proportionally dense subgraph (PDS)* if for every $v \in S$, we have $\frac{|N_G(v) \cap S|}{|S|-1} \geq \frac{|N_G(v) \cap \overline{S}|}{|\overline{S}|}$. Then, the MAX PDS problem consists in finding a proportionally dense subgraph in G of maximum size. The authors of [3] showed that the MAX PDS problem is NP-hard, even when restricted to split graphs or bipartite graphs, and that it can be solved in linear time in cubic Hamiltonian graphs.

By proceeding in a similar way as before, where in the associated auxiliary problem we have only two colors, S and $\overline{\text{S}}$, and the check function is given by $check(v, a, n_0, n_1) = \left(a = \text{S} \Rightarrow \frac{n_{\text{S}}}{s_{\text{S}} - 1} \geq \frac{n_{\overline{\text{S}}}}{s_{\overline{\text{S}}}}\right)$, we can show that MAX PDS is XP parameterized by clique-width.

Another variation defined in [3] is the PDS EXTENSION problem, which asks whether there exists a proportionally dense subgraph $G[S]$ such that $U \subset S$, for some $U \subset V(G)$ given as an input. It was shown in [3] that the PDS EXTENSION problem is NP-complete, and no polynomial time algorithms were known for any graph class. We can show that this problem is also XP parameterized by clique-width, by proceeding almost exactly as explained above, where the only change is that we now set $L_v = \{\text{S}\}$ for all $v \in U$.

Given a graph G and a real number $\gamma \in (0, 1]$, a *degree-based γ-quasi-clique* is defined as a subset $S \subseteq V(G)$ such that the degree of any vertex in $G[S]$ is at least $\gamma(|S| - 1)$, that is, $\frac{|N_G(v) \cap S|}{|S|-1} \geq \gamma$. The MAXIMUM DEGREE-BASED γ-QUASI-CLIQUE problem consists in finding a degree-based γ-quasi-clique of maximum cardinality in a graph. In [24], it was shown that this problem is NP-hard for any fixed γ. Using the same techniques as for MAX-PDS (only slightly modifying the check function), we obtain that MAXIMUM DEGREE-BASED γ-QUASI-CLIQUE is XP parameterized by clique-width.

References

1 H. Abdollahzadeh Ahangar, M.P. Álvarez, M. Chellali, S.M. Sheikholeslami, and J.C. Valenzuela-Tripodoro. Triple roman domination in graphs. *Applied Mathematics and Computation*, 391:125444, 2021. doi:10.1016/j.amc.2020.125444.

2 Cristina Bazgan, Janka Chlebikova, and Thomas Pontoizeau. Structural and algorithmic properties of 2-community structures. *Algorithmica*, 80:1890–1908, 2018. doi:10.1007/s00453-017-0283-7.

3 Cristina Bazgan, Janka Chlebíková, Clément Dallard, and Thomas Pontoizeau. Proportionally dense subgraph of maximum size: Complexity and approximation. *Discrete Applied Mathematics*, 270:25–36, 2019. doi:10.1016/j.dam.2019.07.010.

4 Robert A. Beeler, Teresa W. Haynes, and Stephen T. Hedetniemi. Double Roman domination. *Discrete Applied Mathematics*, 211:23–29, 2016. doi:10.1016/j.dam.2016.03.017.

5 Benjamin Bergougnoux, Jan Dreier, and Lars Jaffke. A logic-based algorithmic meta-theorem for mim-width, 2022. doi:10.48550/arXiv.2202.13335.

6 Flavia Bonomo-Braberman and Carolina Lucía Gonzalez. A new approach on locally checkable problems. *Discrete Applied Mathematics*, 314:53–80, 2022. doi:10.1016/j.dam.2022.01.019.

7 Binh-Minh Bui-Xuan, Jan Arne Telle, and Martin Vatshelle. Fast dynamic programming for locally checkable vertex subset and vertex partitioning problems. *Theoretical Computer Science*, 511:66–76, 2013. doi:10.1016/j.tcs.2013.01.009.

8 Yair Caro and Michael S. Jacobson. On non-z(mod k) dominating sets. *Discussiones Mathematicae Graph Theory*, 23(1):189–199, 2003. doi:10.7151/dmgt.1195.

9 Ernie J Cockayne, Paul A Dreyer, Sandra M Hedetniemi, and Stephen T Hedetniemi. Roman domination in graphs. *Discrete Mathematics*, 278(1):11–22, 2004. doi:10.1016/j.disc.2003.06.004.

10 B. Courcelle, J. Makowsky, and U. Rotics. Linear time solvable optimization problems on graphs of bounded clique-width. *Theory Comput. Systems*, 33:125–150, 2000. doi:10.1007/s002249910009.

11 Bruno Courcelle, Joost Engelfriet, and Grzegorz Rozenberg. Handle-rewriting hypergraph grammars. *Journal of Computer and System Sciences*, 46(2):218–270, 1993. doi:10.1016/0022-0000(93)90004-G.

12 Bruno Courcelle and Stephan Olariu. Upper bounds to the clique width of graphs. *Discrete Applied Mathematics*, 101(1):77–114, 2000. doi:10.1016/S0166-218X(99)00184-5.

13 Vladimir Estivill-Castro and Mahdi Parsa. Hardness and tractability of detecting connected communities. In *Proceedings of the Australasian Computer Science Week Multiconference*, ACSW '16, New York, NY, USA, 2016. Association for Computing Machinery. doi:10.1145/2843043.2843053.

14 Markus Frick and Martin Grohe. The complexity of first-order and monadic second-order logic revisited. *Annals of Pure and Applied Logic*, 130(1):3–31, 2004. Papers presented at the 2002 IEEE Symposium on Logic in Computer Science (LICS). doi:10.1016/j.apal.2004.01.007.

15 Elisabeth Gassner and Johannes Hatzl. A parity domination problem in graphs with bounded treewidth and distance-hereditary graphs. *Computing*, 82:171–187, 2008. doi:10.1007/s00607-008-0005-8.

16 Michael U. Gerber and Daniel Kobler. Algorithms for vertex-partitioning problems on graphs with fixed clique-width. *Theoretical Computer Science*, 299:719–734, 2003.

17 Carolina Lucía Gonzalez and Felix Mann. On d-stable locally checkable problems on bounded mim-width graphs, 2022. doi:10.48550/arXiv.2203.15724.

18 T.W. Haynes, S. Hedetniemi, and P. Slater. *Fundamentals of Domination in Graphs*. CRC Press, 1st edition, 1998. doi:10.1201/9781482246582.

19 John E. Hopcroft and Jeffrey D. Ullman. *Introduction To Automata Theory, Languages, And Computation*. Addison-Wesley Longman Publishing Co., Inc., USA, 1st edition, 1990.

20 Lars Jaffke, O-joung Kwon, Torstein J. F. Strømme, and Jan Arne Telle. Generalized distance domination problems and their complexity on graphs of bounded mim-width, 2018. doi:10.48550/arXiv.1803.03514.

21 F. Nahani Pour, H. Abdollahzadeh Ahangar, M. Chellali, and S.M. Sheikholeslami. Global triple roman dominating function. *Discrete Applied Mathematics*, 314:228–237, 2022.

22 Martin Olsen. A general view on computing communities. *Mathematical Social Sciences*, 66(3):331–336, 2013. doi:10.1016/j.mathsocsci.2013.07.002.

23 Sang-il Oum and Paul Seymour. Approximating clique-width and branch-width. *Journal of Combinatorial Theory, Series B*, 96(4):514–528, 2006. doi:10.1016/j.jctb.2005.10.006.

24 Grigory Pastukhov, Alexander Veremyev, Vladimir Boginski, and Oleg A. Prokopyev. On maximum degree-based -quasi-clique problem: Complexity and exact approaches. *Networks*, 71(2):136–152, 2018. doi:10.1002/net.21791.

25 P. Roushini Leely Pushpam and S. Padmapriea. Global roman domination in graphs. *Discrete Applied Mathematics*, 200:176–185, 2016.

26 Zehui Shao, S. M. Sheikholeslami, S. Nazari-Moghaddam, and Shaohui Wang. Global double roman domination in graphs. *Journal of Discrete Mathematical Sciences and Cryptography*, 22(1):31–44, 2019.

27 Jan Arne Telle and Andrzej Proskurowski. Algorithms for vertex partitioning problems on partial k-trees. *SIAM Journal on Discrete Mathematics*, 10(4):529–550, 1997. doi:10.1137/S0895480194275825.

28 Douglas B. West. *Introduction to Graph Theory*. Prentice Hall, 2001.

A Examples and omitted lemmas

We include here examples of color-counting 1-locally checkable problems, and the lemmas omitted from Section 5. Due to space constraints, only some of the proofs are presented.

A.1 Examples of color-counting 1-locally checkable problems

▶ **Example 14.** Consider the k-Coloring problem. This problem can be seen as a color-counting 1-locally checkable problem with the following characteristics:

- $\textsc{Colors} = [\![1, k]\!]$ and $L_v = [\![1, k]\!]$ for all $v \in V(G)$;
- $(\textsc{Weights}, \preceq, \circledast) = (\{0, 1\}, \leq, \max)$ and $\mathrm{w}_{v,a} = 0$ for all $v \in V(G), a \in L_v$;
- $\mathit{check}(v, a, n_1, \ldots, n_k) = (n_a = 0)$.

▶ **Example 15.** The Maximum Independent Set problem can also be modeled as a color-counting 1-locally checkable problem:

- $\textsc{Colors} = \{0, 1\}$ and $L_v = \{0, 1\}$ for all $v \in V(G)$;
- $(\textsc{Weights}, \preceq, \circledast) = (\mathbb{N} \cup \{-\infty\}, \geq, +)$ and $\mathrm{w}_{v,a} = a$ for all $v \in V(G), a \in L_v$;
- $\mathit{check}(v, a, n_0, n_1) = (a = 0 \vee n_1 = 0)$.

▶ **Example 16.** The Minimum Odd Dominating Set problem can as well be modeled as a color-counting 1-locally checkable problem, as follows:

- $\textsc{Colors} = \{0, 1\}$ and $L_v = \{0, 1\}$ for all $v \in V(G)$;
- $(\textsc{Weights}, \preceq, \circledast) = (\mathbb{N} \cup \{+\infty\}, \leq, +)$ and $\mathrm{w}_{v,a} = a$ for all $v \in V(G), a \in L_v$;
- $\mathit{check}(v, a, n_0, n_1) = (a + n_1 \equiv 1 \mod 2)$.

A.2 Omitted lemmas of Section 5

In what follows, we will write $\widehat{p}$ instead of $p_1, \ldots, p_m$ to make the notation less cumbersome. Note that $\widehat{p}$ is empty when $m = 0$.

▶ **Lemma 17.** *Let Π be color-counting 1-locally checkable problem with a set of global properties Γ, input graph G with a clique-width k-expression e_G of G, and let $a \in$ Colors. Let $\sigma \subseteq \mathbb{N}$ and let $(Q, \{1\}, \delta, q_0, F)$ be a deterministic finite-state automaton that accepts a string of n consecutive 1's if and only if $n \in \sigma$. Let $\in_F \colon Q \to$ Bool be the function such that $\in_F (q) = (q \in F)$.*

Assume that the minimum weight of a proper coloring of G satisfying Γ equals

$$\min\{\lambda(e_G, C, N, \widehat{p}) : P(C, N, \widehat{p}) = \textsc{True}\}$$

for some property P. Furthermore, assume that

(1) *for every proper coloring c of G satisfying Γ there exist $C, N, \widehat{p}$ such that $P(C, N, \widehat{p}) =$ True and such that c is a $(C, N, \widehat{p})$-coloring of G;*

(2) *for every $C, N, \widehat{p}$ such that $P(C, N, \widehat{p}) =$ True and such that $\lambda(e_G, C, N, \widehat{p}) \neq$ Error, every $(C, N, \widehat{p})$-coloring of G is a proper coloring of G satisfying Γ.*

Then, the minimum weight of a proper coloring c of G satisfying Γ and such that $|\{v \in V(G) : c(v) = a\}| \in \sigma$ equals

$$\min\{\lambda(e_G, C, N, \widehat{p}, q_0, \in_F) : P(C, N, \widehat{p}) = \textsc{True}\}.$$

Moreover,

(a) *for every proper coloring c of G satisfying Γ and such that $|\{v \in V(G) : c(v) = a\}| \in \sigma$, there exist $C, N, \widehat{p}$ such that $P(C, N, \widehat{p}) =$ True and such that c is $(C, N, \widehat{p}, q_0, \in_F)$-coloring of G,*

(b) *for every $C, N, \widehat{p}$ such that $P(C, N, \widehat{p}) =$ True and such that $\lambda(e_G, C, N, \widehat{p}, q_0, \in_F) \neq$ Error, every $(C, N, \widehat{p}, q_0, \in_F)$-coloring c of G is a proper coloring of G satisfying Γ and such that $|\{v \in V(G) : c(v) = a\}| \in \sigma$.*

Proof. We will first show that for every proper coloring c of G satisfying Γ and such that $|\{v \in V(G) : c(v) = a\}| \in \sigma$, there exist $C, N, \widehat{p}$ such that $P(C, N, \widehat{p}) =$ True and such that $\mathrm{w}(c) \geq \lambda(e_G, C, N, \widehat{p}, q_0, \in_F)$. Suppose we have such a proper coloring c of G satisfying Γ and such that $|\{v \in V(G) : c(v) = a\}| \in \sigma$. By assumption (1), we know that there exist $C, N, \widehat{p}$ such that $P(C, N, \widehat{p}) =$ True and c is a $(C, N, \widehat{p})$-coloring of G. Since we are assuming that $|\{v \in V(G) : c(v) = a\}| \in \sigma$, and since the automaton accepts a string of t consecutive 1's if and only if $t \in \sigma$, it follows that $\in_F (\delta^n(q_0)) =$ True, where $n = |\{v \in V(G) : c(v) = a\}|$. Thus, c is a $(C, N, \widehat{p}, q_0, \in_F)$-coloring of G and $\mathrm{w}(c) \geq \lambda(C, N, \widehat{p}, q_0, \in_F)$.

On the other hand, we will now show that for every $C, N, \widehat{p}$ such that $P(C, N, \widehat{p}) =$ True and such that $\lambda(e_G, C, N, \widehat{p}, q_0, \in_F) \neq$ Error, there exists a proper coloring c of G satisfying Γ and such that $|\{v \in V(G) : c(v) = a\}| \in \sigma$ with $\mathrm{w}(c) = \lambda(e_G, C, N, \widehat{p}, q_0, \in_F)$. So suppose we have $C, N, \widehat{p}$ such that $P(C, N, \widehat{p}) =$ True and such that $\lambda(C, N, \widehat{p}, q_0, \in_F) \neq$ Error. By the latter assumption and by the definition of $\lambda(C, N, \widehat{p}, q_0, \in_F)$, we get that $\lambda(C, N, \widehat{p}) \neq$ Error. Let c be a $(C, N, \widehat{p}, q_0, \in_F)$-coloring of G (notice that at least one such c exists). By definition, c is a $(C, N, \widehat{p})$-coloring of G. Then we conclude by assumption (2) that c is a proper coloring of G satisfying Γ. Finally, since c is a $(C, N, \widehat{p}, q_0, \in_F)$-coloring of G, we have that $\in_F (\delta^n(q_0)) =$ True, where $n = |\{v \in V(G) : c(v) = a\}|$, which implies that $|\{v \in V(G) : c(v) = a\}| \in \sigma$. If we consider in particular c of minimum weight, then $\mathrm{w}(c) = \lambda(e_G, C, N, \widehat{p}, q_0, \in_F)$.

Notice that (a) and (b) are implicitly shown by the above. ◀

▶ **Lemma 18** (Creating new vertex: $i(v)$)**.** *If $C[i,a] = 1$ and $f_a(\delta(s_a,1)) = \textsc{True}$, or if $C[i,a] = 0$ and $f_a(s_a) = \textsc{True}$, then*

$$\lambda(i(v), C, N, \widehat{p}, s_a, f_a) = \lambda(i(v), C, N, \widehat{p}).$$

Otherwise, $\lambda(i(v), C, N, \widehat{p}, s_a, f_a) = \textsc{Error}$.

▶ **Lemma 19** (Disjoint union: $e_1 \oplus e_2$)**.** *Assume that*

$$\lambda(e_1 \oplus e_2, C, N, \widehat{p}) = \min\{\lambda(e_1, C_1, N_1, \widehat{p_1}) \circledast \lambda(e_2, C_2, N_2, \widehat{p_2}) : P(C, N, \widehat{p}, C_1, N_1, \widehat{p_1}, C_2, N_2, \widehat{p_2}) = \textsc{True}\}$$

for some property P. Moreover, assume that

(1) *for every $(C, N, \widehat{p})$-coloring c of $G_{e_1 \oplus e_2}$ there exist $C_1, N_1, \widehat{p_1}, C_2, N_2, \widehat{p_2}$ such that $P(C, N, \widehat{p}, C_1, N_1, \widehat{p_1}, C_2, N_2, \widehat{p_2}) = \textsc{True}$ and such that $c|_{V(G_{e_1})}$ is a $(C_1, N_1, \widehat{p_1})$-coloring of G_{e_1} and $c|_{V(G_{e_2})}$ is a $(C_2, N_2, \widehat{p_2})$-coloring of G_{e_2};*

(2) *for all $C_1, N_1, \widehat{p_1}, C_2, N_2, \widehat{p_2}$ such that $P(C, N, \widehat{p}, C_1, N_1, \widehat{p_1}, C_2, N_2, \widehat{p_2}) = \textsc{True}$, if c_1 is a $(C_1, N_1, \widehat{p_1})$-coloring of G_{e_1} and c_2 is a $(C_2, N_2, \widehat{p_2})$-coloring of G_{e_2}, then $c = c_1 \cup c_2$ is a $(C, N, \widehat{p})$-coloring of $G_{e_1 \oplus e_2}$.*

Then,

$$\lambda(e_1 \oplus e_2, C, N, \widehat{p}, s_a, f_a) = \min\{\lambda(e_1, C_1, N_1, \widehat{p_1}, s_a, eq_q) \circledast \lambda(e_2, C_2, N_2, \widehat{p_2}, q, f_a) : q \in Q \text{ and } P(C, N, \widehat{p}, C_1, N_1, \widehat{p_1}, C_2, N_2, \widehat{p_2}) = \textsc{True}\}.$$

Moreover,

(a) *for every $(C, N, \widehat{p}, s_a, f_a)$-coloring c of $G_{e_1 \oplus e_2}$ there exist q, $C_1, N_1, \widehat{p_1}, C_2, N_2, \widehat{p_2}$ such that $q \in Q$, $P(C, N, \widehat{p}, C_1, N_1, \widehat{p_1}, C_2, N_2, \widehat{p_2}) = \textsc{True}$, and $c|_{V(G_{e_1})}$ is a $(C_1, N_1, \widehat{p_1}, s_a, eq_q)$-coloring of G_{e_1} and $c|_{V(G_{e_2})}$ is a $(C_2, N_2, \widehat{p_2}, q, f_a)$-coloring of G_{e_2};*

(b) *for all q, $C_1, N_1, \widehat{p_1}, C_2, N_2, \widehat{p_2}$ such that $q \in Q$ and $P(C, N, \widehat{p}, C_1, N_1, \widehat{p_1}, C_2, N_2, \widehat{p_2}) = \textsc{True}$, if c_1 is a $(C_1, N_1, \widehat{p_1}, s_a, eq_q)$-coloring of G_{e_1} and c_2 is a $(C_2, N_2, \widehat{p_2}, q, f_a)$-coloring of G_{e_2} then $c = c_1 \cup c_2$ is a $(C, N, \widehat{p}, s_a, f_a)$-coloring of $G_{e_1 \oplus e_2}$.*

▶ **Lemma 20** (Join: $\eta_{i,j}(e)$)**.** *Assume that there exist $C', N', \widehat{p'}$ such that c is a $(C, N, \widehat{p})$-coloring of $G_{\eta_{i,j}(e)}$ if and only if c is a $(C', N', \widehat{p'})$-coloring of G_e.*

Then, c is a $(C, N, \widehat{p}, s_a, f_a)$-coloring of $G_{\eta_{i,j}(e)}$ if and only if c is a $(C', N', \widehat{p'}, s_a, f_a)$-coloring of G_e. In particular,

$$\lambda(\eta_{i,j}(e), C, N, \widehat{p}, s_a, f_a) = \lambda(e, C', N', \widehat{p'}, s_a, f_a).$$

▶ **Lemma 21** (Relabeling: $\rho_{i \to j}(e)$)**.** *Assume that*

$$\lambda(\rho_{i \to j}(e), C, N, \widehat{p}) = \min\{\lambda(e, C_e, N_e, \widehat{p_e}) : P(C, N, \widehat{p}, C_e, N_e, \widehat{p_e}) = \textsc{True}\}$$

for some property P. Moreover, assume that

(1) *for every $(C, N, \widehat{p})$-coloring c of $G_{\rho_{i \to j}(e)}$, there exist parameters $C_e, N_e, \widehat{p_e}$ such that $P(C, N, \widehat{p}, C_e, N_e, \widehat{p_e}) = \textsc{True}$ and c is a $(C_e, N_e, \widehat{p_e})$-coloring of G_e;*

(2) *for all parameters $C_e, N_e, \widehat{p_e}$ such that $P(C, N, \widehat{p}, C_e, N_e, \widehat{p_e}) = \textsc{True}$, if c is a $(C_e, N_e, \widehat{p_e})$-coloring of G_e, then c is also a $(C, N, \widehat{p})$-coloring c of $G_{\rho_{i \to j}(e)}$.*

Then,

$$\lambda(\rho_{i\to j}(e), C, N, \widehat{p}, s_a, f_a) = \min\{\lambda(e, C_e, N_e, \widehat{p_e}, s_a, f_a) : P(C, N, \widehat{p}, C_e, N_e, \widehat{p_e}) = \textsc{True}\}.$$

Moreover,

(a) *for every $(C, N, \widehat{p}, s_a, f_a)$-coloring c of $G_{\rho_{i\to j}(e)}$, there exist parameters $C_e, N_e, \widehat{p_e}$ such that $P(C, N, \widehat{p}, C_e, N_e, \widehat{p_e}) = \textsc{True}$ and c is a $(C_e, N_e, \widehat{p_e}, s_a, f_a)$-coloring of G_e;*

(b) *for all parameters $C_e, N_e, \widehat{p_e}$ such that $P(C, N, \widehat{p}, C_e, N_e, \widehat{p_e}) = \textsc{True}$, if c is a $(C_e, N_e, \widehat{p_e}, s_a, f_a)$-coloring of G_e, then c is also a $(C, N, \widehat{p}, s_a, f_a)$-coloring c of $G_{\rho_{i\to j}(e)}$.*

Proof. Let $\alpha = \min\{\lambda(e, C_e, N_e, \widehat{p_e}, s_a, f_a) : P(C, N, \widehat{p}, C_e, N_e, \widehat{p_e}) = \textsc{True}\}$. We will first prove that $\lambda(\rho_{i\to j}(e), C, N, \widehat{p}, s_a, f_a) \geq \alpha$. Let c be a $(C, N, \widehat{p}, s_a, f_a)$-coloring of $G_{\rho_{i\to j}(e)}$. We show that there exist parameters $C_e, N_e, \widehat{p_e}$ such that $P(C, N, \widehat{p}, C_e, N_e, \widehat{p_e}) = \textsc{True}$ and c is a $(C_e, N_e, \widehat{p_e}, s_a, f_a)$-coloring of G_e. By definition, c is a $(C, N, \widehat{p})$-coloring of $G_{\rho_{i\to j}(e)}$ such that $f_a(\delta^n(s_a)) = \textsc{True}$, where $n = |\{v \in V(G_{\rho_{i\to j}(e)}) : c(v) = a\}|$. Therefore, by assumption (1), there exist parameters $C_e, N_e, \widehat{p_e}$ such that $P(C, N, \widehat{p}, C_e, N_e, \widehat{p_e}) = \textsc{True}$ and c is a $(C_e, N_e, \widehat{p_e})$-coloring of G_e. Furthermore, since $n_e = |\{v \in V(G_e) : c(v) = a\}| = |\{v \in V(G_{\rho_{i\to j}(e)}) : c(v) = a\}| = n$, it immediately follows that $f_a(\delta^{n_e}(s_a)) = \textsc{True}$. Thus, c is a $(C_e, N_e, \widehat{p_e}, s_a, f_a)$-coloring of G_e. If we consider in particular a $(C, N, \widehat{p}, s_a, f_a)$-coloring c of $G_{\rho_{i\to j}(e)}$ of minimum weight, then $\lambda(\rho_{i\to j}(e), C, N, \widehat{p}, s_a, f_a) = \mathrm{w}(c) \geq \lambda(e, C_e, N_e, \widehat{p_e}, s_a, f_a) \geq \alpha$.

Let us now show that $\lambda(\rho_{i\to j}(e), C, N, \widehat{p}, s_a, f_a) \leq \alpha$. Let $C_e, N_e, \widehat{p_e}$ be such that $P(C, N, \widehat{p}, C_e, N_e, \widehat{p_e}) = \textsc{True}$. Let c be a $(C_e, N_e, \widehat{p_e}, s_a, f_a)$-coloring of G_e. By definition, c is a $(C_e, N_e, \widehat{p_e})$-coloring of G_e such that $f_a(\delta^{n_e}(s_a)) = \textsc{True}$, where $n_e = |\{v \in V(G_e) : c(v) = a\}|$. Then, by assumption (2), c is also a $(C, N, \widehat{p})$-coloring c of $G_{\rho_{i\to j}(e)}$. Furthermore, since $n = |\{v \in V(G_{\rho_{i\to j}(e)}) : c(v) = a\}| = |\{v \in V(G_e) : c(v) = a\}| = n_e$, it immediately follows that $f_a(\delta^n(s_a)) = \textsc{True}$. Therefore, c is a $(C, N, \widehat{p}, s_a, f_a)$-coloring c of $G_{\rho_{i\to j}(e)}$. If we consider in particular a $(C_e, N_e, \widehat{p_e}, s_a, f_a)$-coloring c of G_e for which α is obtained, then $\alpha = \mathrm{w}(c) \geq \lambda(\rho_{i\to j}(e), C, N, \widehat{p}, s_a, f_a)$.

Notice that (a) and (b) are implicitly shown by the above. ◀

A.2.1 Complexity of the modified algorithm

First, assume that for any possible parameter f_a and state q, $\delta(s, 1)$ and $f_a(q)$ can be computed in constant time. Also, assume that we want to fix the size of $\mathcal{R}$ color classes $a_1, \ldots, a_{\mathcal{R}}$. Let $\mathcal{S}$ be the size of the largest set of states among the $\mathcal{R}$ considered automata. Then, we add a term $\mathcal{R}$ to the complexity corresponding to the operation of creating a new labeled vertex, and we multiply by a term $\mathcal{S}^{\mathcal{R}}$ the complexity corresponding to the disjoint union operation. Moreover, whenever we go through all the possible $\lambda(e, C, N, \widehat{p}, s_{a_1}, f_{a_1}, \ldots, s_{a_{\mathcal{R}}}, f_{a_{\mathcal{R}}})$, we multiply the complexity by a factor $(\mathcal{S}(\mathcal{S}+1))^{\mathcal{R}}$. Hence, since $\mathcal{R}$ is at most the number of colors and $\mathcal{S} \leq |V(G)|$, we conclude that the new algorithm is also XP parameterized by clique-width when the number of colors is constant.