



HAL
open science

Improved Bounds for Twin-Width Parameter Variants with Algorithmic Applications to Counting Graph Colorings

Ambroise Baril, Miguel Couceiro, Victor Lagerkvist

► **To cite this version:**

Ambroise Baril, Miguel Couceiro, Victor Lagerkvist. Improved Bounds for Twin-Width Parameter Variants with Algorithmic Applications to Counting Graph Colorings. 2023. hal-04142719v1

HAL Id: hal-04142719

<https://hal.science/hal-04142719v1>

Preprint submitted on 27 Jun 2023 (v1), last revised 13 May 2024 (v3)

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution 4.0 International License

1 Linear Bounds between Component Twin-Width 2 and Clique-Width with Algorithmic Applications to 3 Counting Graph Colorings

4 Ambroise Baril ✉

5 LORIA, Université de Lorraine, France

6 Miguel Couceiro ✉

7 LORIA, Université de Lorraine, France

8 Victor Lagerkvist ✉

9 Linköping Universitet, Sweden

10 — Abstract —

11 The H -COLORING problem is a well-known generalization of the classical NP-complete problem
12 k -COLORING where the task is to determine whether an input graph admits a homomorphism to the
13 template graph H . This problem has been the subject of intense theoretical research and in this
14 paper we study the complexity of H -COLORING with respect to the parameters *clique-width* and the
15 more recent *component twin-width*, which describe desirable computational properties of graphs. We
16 give two surprising linear bounds between these parameters, thus improving the previously known
17 exponential and double exponential bounds. These linear bounds entail natural approximations of
18 component twin-width, by making use of the results known for clique-width. On the algorithmic side
19 we target the richer problem of counting the number of homomorphism to H ($\#H$ -COLORING). The
20 first algorithm uses a contraction sequence of the input graph G parameterized by the component
21 twin-width of G . This leads to a positive FPT result for the counting version. The second uses a
22 contraction sequence of the template graph H and here we instead measure the complexity with
23 respect to the number of vertices in the input graph. Using our linear bounds we show that the
24 latter *always* beats the previously best algorithm (based on clique-width) and we thus obtain the
25 fastest general purpose algorithm for graph coloring.

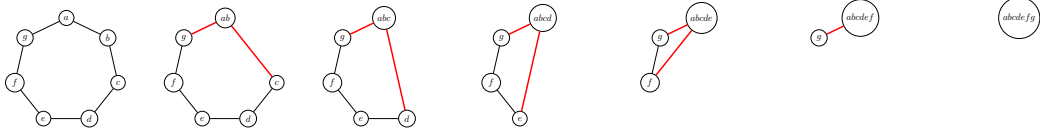
26 **2012 ACM Subject Classification** Discrete mathematics

27 **Keywords and phrases** Component twin-width, Clique-width, Graph coloring, Parameterized com-
28 plexity, Fine-grained complexity

29 **1** Introduction

30 *Graph coloring* is a well-known computational problem where the goal is to color a graph
31 in a consistent way. This problem is one of the most well-studied NP-hard problems and
32 enjoys a wide range of applications *e.g.*, in planning, scheduling, and resource allocation [20].
33 There are many variants and different formulations of the coloring problem, but the most
34 common formulation is certainly the k -COLORING problem that asks whether the vertices of
35 an input graph can be colored using k available colors in such a way that no two adjacent
36 vertices are assigned the same color. This problem can be extended in many ways and in this
37 paper we are particularly interested in the more general problem where any two adjacent
38 vertices in the input graph G have to be mapped to two adjacent vertices in a fixed template
39 graph H (the H -COLORING problem). It is not difficult to see that k -COLORING is then
40 K_k -COLORING, where K_k is the k -vertex clique.

41 The basic H -COLORING problem has been extended in many directions, of which one
42 of the most dominant formalisms is the *counting* extension of where the task is not only to
43 decide whether there is at least one solution (coloring) but to return the number of solutions
44 ($\#H$ -COLORING). This framework makes it possible to encode phase transition systems



■ **Figure 1** A contraction sequence of the 7-cycle.

45 modelled by partition functions, modeling problems from statistical physics such as counting
 46 q -particle Widom–Rowlinson configurations and counting Beach models, or the classical Ising
 47 model (for further examples, see *e.g.* Dyer & Greenhill [17]). The $\#H$ -COLORING problem is
 48 $\#P$ -hard unless every connected component of H is either a single vertex without a loop,
 49 a clique or a bipartite complete graph, and it is in P otherwise [17]. The question is then
 50 to which degree we can still hope to solve it efficiently, or at least improve upon the naive
 51 bound of $|V_H|^{|V_G|}$ (where V_H is the set of vertices in the template graph H and V_G the set
 52 of vertices in the input graph G).

53 In this paper we tackle this question by targeting properties of graphs, so-called *graph*
 54 *parameters*, which give rise to efficiently solvable subproblems. We will see below several
 55 concrete examples of graph parameters, but for the moment we simply assume that each
 56 graph G is associated with a number $k \in \mathbb{N}$, a *parameter*, which describes a structural
 57 property of G . Here, the idea is that small values of k correspond to graphs with a simple
 58 structure, while large values correspond to more complicated graphs.

59 There are then two ways to approach intractable H -COLORING problems: we either
 60 restrict the class of *input* graphs G , or the class of *template* graphs H to graphs where
 61 the parameter is bounded by some reasonably small constant. The first task is typically
 62 studied using tools from *parameterized* complexity where goal is to prove that problems are
 63 *fixed-parameter tractable* (FPT), *i.e.*, obtaining running times of the form $f(k) \cdot \|G\|^{O(1)}$ for
 64 a computable function $f: \mathbb{N} \rightarrow \mathbb{N}$ (where $\|G\|$ is the number of bits required to represent the
 65 input graph G). The second task is more closely related to *fine-grained* complexity where
 66 the goal is to prove upper and lower bounds of the form $2^{f(k)} \cdot \|G\|^{O(1)}$ for a sufficiently
 67 “fine-grained” parameter k , which in our case is always going to denote the number of vertices
 68 $|V_G|$ in the input graph G . Here, it is worth remarking that H -COLORING is believed to be a
 69 very hard problem, and the general COLORING problem, where the template is part of the
 70 input, is strongly believed to be unsolvable in $2^{O(|V_G|)} \cdot (\|G\| + \|H\|)^{O(1)}$ time [19]. Hence,
 71 regardless whether one studies the problem under the lens of parameterized or fine-grained
 72 complexity, one needs to limit the class of considered graphs via a suitable parameter.

73 There are several graph parameters proposed to address the limitations of tree-width and
 74 that generalize the class of co-graphs. We briefly survey two noteworthy graph parameters
 75 (see Section 2 for formal definitions).

- 76 1. *clique-width* (**cw**). The class of graphs (with labelled vertices) with clique-width $k \geq 1$
 77 is defined as the smallest class of graphs that contains the one vertex graphs \bullet_i with 1
 78 vertex labelled by $i \in [k]$, and that is stable by the following operations for $(i, j) \in [k]^2$
 79 with $i \neq j$: (i) disjoint union of graphs, (ii) relabelling every vertex of label i to label j ,
 80 and (iii) constructing edges between every vertex labelled by i and every vertex labelled
 81 by j . Note that the class of cographs (which contains cliques) is exactly that of graphs
 82 with clique-width at most 2.
- 83 2. *twin-width* (**tw**). The class of graphs of twin-width $k \geq 1$ is usually formulated via
 84 *contraction sequences* where graphs are gradually merged into a single vertex (see Figure 1
 85 for an example). Red edges represent an inconsistency in the merged vertex (see Section

86 2.3 for a formal definition), and the maximum red degree in the sequence thus represents
 87 the largest loss of information. A graph has twin-width $\leq k$ if it admits such a contraction
 88 sequence where the maximum red degree does not exceed k .

89 For clique-width, Ganian et. al [21] identified a structural parameter s of graphs and
 90 presented an algorithm for H -COLORING that runs in $O^*(s(H)^{\mathbf{cw}(G)})$ time¹. It is also optimal
 91 in the sense that if there exists an algorithm that solves H -COLORING in time $O^*((s(H) -$
 92 $\varepsilon)^{\mathbf{cw}(G)})$, then the SETH fails [21]. Again, alternative algorithms exist for templates of
 93 bounded clique-width, and Wahlström [27] solves $\#H$ -COLORING in $O^*((2\mathbf{cw}(H) + 1)^{|V_G|})$
 94 time.

95 Twin-width, on the other hand, is a much more recent parameter, but has in only a
 96 few years attracted significant attention [1, 2, 3, 5, 4, 6, 7, 8, 9, 11, 12]. One of its greatest
 97 achievement is that checking if a graph is a model of any first order formula can be decided
 98 in FPT time parameterized by the twin-width of the input graph. Thus, a very natural
 99 research question in light of the above results concerning tree- and clique-width is to study
 100 the complexity of ($\#$) H -COLORING via twin-width. Unfortunately, it is easy to see that under
 101 standard assumptions, H -COLORING is generally not FPT parameterized by twin-width.
 102 Indeed, since twin-width is bounded on planar graphs [23], the existence of an FPT algorithm
 103 for 3-COLORING running in $O^*(f(\mathbf{tww}(G)))$ time implies an $O^*(1)$ time (ie. a polynomial
 104 time) algorithm for 3-COLORING on planar graphs (since $f(\mathbf{tww}(G)) = O(1)$ if G is a planar
 105 graph). Since 3-COLORING is NP-hard on planar graphs, this would imply P=NP. Thus,
 106 3-COLORING is *para-NP-hard* [16] parameterized by twin-width.

107 Despite this hardness result it is possible to analyse H -COLORING by a variant of twin-
 108 width known as *component twin-width* (\mathbf{ctww}) [8]. This parameter equals the maximal size
 109 of a red-connected component (instead of the maximal red-degree for twin-width). It is then
 110 known that component twin-width is functionally equivalent² to boolean-width [8], which in
 111 turn is functionally equivalent to clique-width [13]. Hence, H -COLORING is FPT parameterized
 112 by component twin-width, and the specific problem k -COLORING is additionally known to
 113 be solvable in $O^*((2^k - 1)^{\mathbf{ctww}(G)})$ time [8]. As remarked Bonnet et al., the theoretical
 114 implications of this particular algorithm are limited due to the aforementioned (under the
 115 SETH) optimal algorithm parameterized by clique-width [21]. However, this still leaves
 116 several gaps in our understanding of component twin-width for H -COLORING and its counting
 117 extension $\#H$ -COLORING.

118 Our paper has three major contributions to bridge these gaps. *Firstly*, the best known
 119 bounds between clique-width and component twin-width are obtained by following the
 120 proof of functional equivalence between component twin-width and boolean-width, and then
 121 between boolean-width and clique-width. We thereby obtain

$$122 \quad \mathbf{ctww} \leq 2^{\mathbf{cw}+1} \quad \text{and} \quad \mathbf{cw} \leq 2^{2^{\mathbf{ctww}}}$$

123 and H -COLORING is thus solvable in $O^*(s(H)^{2^{2^{\mathbf{ctww}(G)}}})$ time. This proves FPT but is clearly
 124 not a practically applicable algorithm and the main question is whether it is possible to
 125 improve this to a single-exponential running time $O^*(2^{O(\mathbf{ctww}(G))})$. (This line of research is
 126 relatively new but of growing importance and has seen several landmark results [15].) We
 127 prove that it is indeed possible by significantly strengthening the bounds between \mathbf{cw} and

¹ The notation O^* means that we ignore polynomial factors.

² *I.e.*, each parameter is bounded by a function of the other.

128 \mathbf{ctww} and obtain the linear bounds

$$129 \quad \mathbf{cw} \leq \mathbf{ctww} + 1 \leq 2\mathbf{cw}.$$

130 Our proof is constructive which gives a fast algorithm to derive a contraction-sequence from
131 a clique-width expression and vice versa.

132 *Secondly*, we discuss how these bounds can be exploited to *approximate* \mathbf{ctww} by making
133 use of the results known on \mathbf{cw} . Thus, an immediate consequence of our linear bounds is
134 that H -COLORING is solvable in $O^*(s(H)^{\mathbf{ctww}(G)+1})$ time, which is a major improvement to
135 the aforementioned triple exponential upper bound.

136 *Thirdly*, we consider the generalized problem of counting the number of solutions. It
137 seems unlikely that the optimal algorithm (under SETH) by Ganian et al. [21] can be
138 lifted to $\#H$ -COLORING, and while the algorithm by Wahlström [27] successfully solves
139 $\#H$ -COLORING, it does so with the significantly worse bound of $O^*((2\mathbf{cw}(H) + 1)^{|V_G|})$. We
140 tackle this problem in Section 5 by designing a novel algorithm for $\#H$ -COLORING for input
141 graphs with bounded component twin-width and which runs in $O^*((2^{|V_H|} - 1)^{\mathbf{ctww}(G)})$ time.

142 We also consider $\#H$ -COLORING when the template graph H has bounded component
143 twin-width. We use an optimal contraction sequence of H in order to obtain a $O^*((\mathbf{ctww}(H) +$
144 $2)^{|V_G|})$ algorithm for $\#H$ -COLORING. Combining this result with our linear bounds $\mathbf{cw} \leq$
145 $\mathbf{ctww} + 1 \leq 2\mathbf{cw}$, we conclude that our algorithm always runs faster (asymptotically) than
146 the $O^*((2\mathbf{cw}(H) + 1)^{|V_G|})$ time algorithm by Wahlström [27], which was, to our knowledge,
147 the fastest general $\#H$ -COLORING algorithm available in the literature. Moreover, the
148 technique employed in this paper could similarly be used to derive the same results applied
149 to the more general frameworks of counting the solutions of *binary constraint satisfaction*
150 *problems*, ie. problems of the forms $\#\text{BINARY-CSP}(\Gamma)$ with Γ a set of binary relations over a
151 finite domain, even though we restrict to $\#H$ -COLORING here for the sake of simplicity.

152 **2 Preliminaries**

153 Throughout this paper, a *graph* G is a tuple (V_G, E_G) , where V_G is a finite set (the set of
154 vertices of G), and E_G is a binary irreflexive symmetric relation over V_G (the set of edges
155 of G). We will denote the number of vertices of a graph G by $n(G)$ or, simply, by n when
156 there is no danger of ambiguity. The neighborhood of a vertex u of a graph G is the set
157 $N_G(u) = \{v \in V_G \mid (u, v) \in E_G\}$. For a graph H we let H -COLORING be the computational
158 problem of deciding whether there exists an homomorphism from an input graph G to
159 H , i.e., whether there exists a function $f: V_G \rightarrow V_H$ such that $(x, y) \in E_G$ implies that
160 $(f(x), f(y)) \in E_H$. We write $\#H$ -COLORING for the associated *counting* problem where
161 we instead wish to determine the exact number of such homomorphisms. As remarked in
162 Section 1, the template graph H can be chosen with great flexibility to model many different
163 types of problems.

164 **2.1 Parameterized complexity**

165 We assume that the reader is familiar with parameterized complexity and only introduce
166 the strictly necessary concepts (we refer to Flum & Grohe [18] for further background).
167 A *parameterized counting problem* is a pair (F, dom) where $F: \Sigma^* \mapsto \mathbb{N}$ (for an alphabet
168 Σ , i.e., a finite set of symbols) and dom is a subset of $\Sigma^* \times \mathbb{N}$. A parameterized counting
169 problem (F, dom) is said to be *fixed-parameter tractable* (FPT) if there exists a computable
170 function $f: \mathbb{N} \rightarrow \mathbb{N}$ such that for any instance $(x, k) \in \text{dom}$ of F , we can compute $F(x)$ in

171 $f(k) \times \|x\|^{O(1)}$ time. An algorithm with this complexity is said to be an FPT *algorithm*.
 172 Note that even though f might be superpolynomial, which is expected if the problem is
 173 NP-hard, instances where k is reasonably small can still be efficiently solved.

174 In practice, when studying FPT algorithms for an NP-hard counting problem, it is very
 175 natural to optimize the superpolynomial function f that appears in the complexity of the
 176 algorithm solving it. Typically, when dealing with graphs problems parameterized by the
 177 number of vertices n , an algorithm running in $c^n \times \|x\|^{O(1)}$ will be considered efficient in
 178 practice if $c > 1$ is small. This field of research is referred to as *fine-grained complexity*.

179 2.2 Clique-width

180 For $k \geq 1$, let $[k] = \{1, \dots, k\}$. A k -labelled graph G is a tuple (V_G, E_G, l_G) , where (V_G, E_G)
 181 is a graph and $l_G : V_G \rightarrow [k]$. For $i \in [k]$ and a k -labelled graph G , denote by $V_G^i = l_G^{-1}(\{i\})$
 182 the set of vertices of G of label i . A k -expression φ of a k -labelled graph G , denoted $[\varphi] = G$,
 183 is an expression defined inductively [14] using:

- 184 1. \bullet_i with $i \in [k]$: $[\bullet_i]$ is a k -labelled graph with one vertex labelled by i ,
- 185 2. $\rho_{i \rightarrow j}(\varphi)$ with $(i, j) \in [k]^2$ and $i \neq j$: $[\rho_{i \rightarrow j}(\varphi)]$ is the same graph as $[\varphi]$, but where all
 186 vertices of G with label i now have label j ,
- 187 3. $\eta_{i,j}(\varphi)$ with $(i, j) \in [k]^2$ and $i \neq j$: $[\eta_{i,j}(\varphi)]$ is the same graph as $[\varphi]$, but where all tuples
 188 of the form (u, v) with $\{l_G(u), l_G(v)\} = \{i, j\}$ is now an edge, and
- 189 4. $\varphi_1 \oplus \varphi_2$: $[\varphi_1 \oplus \varphi_2]$ is the disjoint union of the graphs $[\varphi_1]$ and $[\varphi_2]$.

190 A graph G has a k -expression φ if there exists $l : V_G \mapsto [k]$ such that $[\varphi] = (V_G, E_G, l)$.
 191 The *clique-width* of a graph G (denoted by $\mathbf{cw}(G)$) is the minimum $k \geq 1$ such that G has a
 192 k -expression. An *optimal expression* of a graph G is a $\mathbf{cw}(G)$ -expression of G .

193 The *subexpressions* of an expression φ are defined similarly: the only subexpression of
 194 \bullet_i is \bullet_i , the subexpressions of $\varphi = \varphi_1 \oplus \varphi_2$ are φ and the subexpressions of φ_1 and φ_2 , the
 195 subexpressions of $\varphi = \rho_{i \rightarrow j}(\varphi')$ and $\varphi = \eta_{i,j}(\varphi')$ are φ and the subexpressions of φ' .

196 A *linear k -expression* is a k -expression φ where for every subexpression of φ of the form
 197 $\varphi_1 \oplus \varphi_2$, φ_2 is of the form \bullet_i with $i \in [k]$. The *linear clique-width* (denoted by $\mathbf{linearcw}(G)$)
 198 of a graph G is the minimum $k \geq 1$ such that G has a linear k -expression.

199 2.3 Component twin-width

200 A *trigraph* is a triplet $G = (V_G, E_G, R_G)$ where (V_G, E_G) and (V_G, R_G) are graphs, and
 201 $E_G \cap R_G \neq \emptyset$. The set V_G is said to be the set of vertices of G , E_G is the set of (black) edges
 202 of G , and R_G the set of *red edges* of G . A red-connected component of a trigraph G is a
 203 connected component of the graph (V_G, R_G) .

204 Let G be a trigraph and $(u, v) \in (V_G)^2$ with $u \neq v$. The trigraph $G/(u, v)$ is defined as
 205 the graph G where u and v have been removed and replaced by a new vertex uv , and where
 206 for all $z \in V_G \setminus \{u, v\}$:

- 207 $\blacksquare (uv, z) \in E_{G/(u,v)}$ if $(u, z) \in E_G$ and $(v, z) \in E_G$,
- 208 $\blacksquare (uv, z) \notin (E_{G/(u,v)} \cup R_{G/(u,v)})$ if $(u, z) \notin (E_G \cup R_G)$ and $(v, z) \notin (E_G \cup R_G)$, and
- 209 $\blacksquare (uv, z) \in R_{G/(u,v)}$ otherwise, *i.e.* when (u, z) or (v, z) is already a red-edge, or when
 210 among (u, z) and (v, z) , one is a black edge and one is a non-edge.

211 A *contraction* of a trigraph G is a graph of the form $G/(u, v)$ with $(u, v) \in (V_G)^2$ and
 212 $u \neq v$. A contraction sequence of a graph G is a sequence of trigraphs of the form (G_n, \dots, G_1)
 213 with $n = |V_G|$, $G_n = (G, \emptyset)$, and for all $k \in [n - 1]$, G_k is a contraction of G_{k+1} . Note that

214 G_k has k vertices and, in particular, that G_1 has only one vertex, which is necessarily the
 215 trigraph³ $G_1 = (\{V_G\}, \emptyset, \emptyset)$. A contraction sequence is said to be *linear* if any of its trigraph
 216 contains at most one red-connected component with at least 2 vertices of the original graph.

217 The *component twin-width* of a contraction sequence (G_n, \dots, G_1) is the maximal size of a
 218 red-connected component among the trigraphs G_k for $k \in [n]$. The (*linear*) *component twin-*
 219 *width* of a graph G denoted by $\mathbf{linearctww}(G)/\mathbf{ctww}(G)$ is the minimum (linear) component
 220 twin-width of all its contraction sequences. The soundness proof of our algorithms that
 221 make use of contraction sequences rely on the following fundamental property of contraction
 222 sequences. This property can be easily proven by induction on k , knowing that it is indeed
 223 true for $G_n = (V_G, E_G, \emptyset)$, and using the definition of a contraction of a trigraph.

224 ► **Property 1.** *Let (G_n, \dots, G_1) be a contraction sequence of a graph G , $k \in [n]$, U and V*
 225 *two different vertices of G_k , $u \in U$ and $v \in V$.³ Then*

- 226 ■ $(u, v) \in E_G$, whenever $(U, V) \in E_{G_k}$, and
- 227 ■ $(u, v) \notin E_G$, whenever $(U, V) \notin E_{G_k} \cup R_{G_k}$.

228 A *cograph* is a graph that has a contraction sequence with no red-edges, *i.e.*, graphs with
 229 component twin-width 1 [10].

2.4 Rank-width

230
 231 A *branch decomposition* of a graph G is a ternary tree T (a tree where each non-leaf vertex has
 232 degree 3) whose set of leaves is exactly V_G . Let G be a graph and T a branch decomposition
 233 of G . Every edge e of T corresponds to a bipartition (X_e, Y_e) of V_G by considering the
 234 bipartition of the leaves of T into their connected components of $T - e$ (the tree T but where
 235 the edge e have been removed). For every edge e of T , let A_e be the $(\mathbb{Z}/2\mathbb{Z})$ -matrix whose set
 236 of rows is X_e and whose set of columns is Y_e , and whose coefficient of index $(u, v) \in X_e \times Y_e$
 237 is 1 if $(u, v) \in E_G$, and 0 otherwise.

238 Finally, let $\rho_G(T) = \max_{e \in E_T} \mathbf{rank}(A_e)$. The *rank-width* of G denoted by $\mathbf{rw}(G)$ is the
 239 minimum of $\rho_G(T)$ for T a branch-decomposition of G . A branch decomposition T realising
 240 this minimum is called an *optimal* branch-decomposition of G . The main interest of rank-
 241 width is made clear in the following remark.

242 ► **Remark 2.** Let T be an optimal branch-decomposition of a graph G , and $e \in E_T$. If $|X_e| >$
 243 $2^{\mathbf{rw}(G)}$, then there exists $(u, u') \in (X_e)^2$ with $u \neq u'$ such that $N_G(u) \cap Y_e = N_G(u') \cap Y_e$.

244 **Proof.** Since the rank of the matrix A_e is lower than $\mathbf{rw}(G)$, the rows of G all belong to a
 245 $(\mathbb{Z}/2\mathbb{Z})$ -vector space of dimension at most $\mathbf{rw}(G)$. The latter has a cardinality of at most
 246 $2^{\mathbf{rw}(G)}$, and therefore, X_e has 2 identical rows, which proves the result. ◀

3 Comparing Clique-Width and Component Twin-Width

248 In this section, we prove the linear bounds between clique-width \mathbf{cw} and component twin-
 249 width \mathbf{ctww} :

$$250 \quad \mathbf{cw} \leq \mathbf{ctww} + 1 \leq 2\mathbf{cw}$$

251 and that the similar bounds hold for their linear versions.

252 Firstly, we prove the leftmost inequality.

³ Each vertex of G_k is a set of vertices of G that have been contracted.

253 ► **Theorem 3.** For every graph G , $\text{cw}(G) \leq \text{ctww}(G) + 1$.

254 **Proof.** Let (G_n, \dots, G_1) be an optimal contraction sequence of G , and let $\kappa = \text{ctww}(G)$.
 255 Note that, for all $k \in [n]$, every red-connected component of G_k has size $\leq \kappa$. We explain
 256 how to construct a $(\kappa + 1)$ -expression of G .

257 We show the following invariant for all $k \in [n]$:

258 $\mathcal{P}(k)$: “Let $C = \{S_1, \dots, S_p\}$ be a red-connected component of G_k and³ $\bigcup C = S_1 \cup \dots \cup S_p$.
 259 There exists a $(\kappa + 1)$ -expression φ_C of the p -labelled graph $G_C = G[\bigcup C]$ with $\forall i \in [p], V_{G_C}^i =$
 260 S_i .”

261 We first prove $\mathcal{P}(n)$. In $G_n = (V_G, E_G, \emptyset)$, there are no red edges: the red-connected
 262 component are the singletons $\{u\}$ for $u \in V_G$. Thus \bullet_1 is a $(\kappa + 1)$ -expression of $(G[\{u\}], l_u)$
 263 (with $l_u : u \mapsto 1$), which proves $\mathcal{P}(n)$.

264 Now, take $k \in [n - 1]$ and assume $\mathcal{P}(k + 1)$. We will prove $\mathcal{P}(k)$. By definition of
 265 a contraction sequence, G_k is of the form $G_k = G_{k+1}/(u, v)$ for two different vertices u
 266 and v of G_{k+1} . Observe that each red-connected component of G_k is also a red-connected
 267 component of G_{k+1} , except the red-connected component C containing uv . Hence, it suffices
 268 to prove $\mathcal{P}(k)$ for the red-connected component C . Notice also that $(C \setminus \{uv\}) \cup \{u, v\}$ is a
 269 union of red-connected component C_1, \dots, C_q of G_{k+1} (every pair of red-connected vertices
 270 in G_{k+1} that does not contain u or v is also red-connected in G_k). We thus have that
 271 $C = (C_1 \cup \dots \cup C_q \cup \{uv\}) \setminus \{u, v\}$.

272 Denote by $\{S_1, \dots, S_{p-1}, S'_p\}$ the set of vertices of C , with $p = |C|$, and $S'_p = uv$. We
 273 have seen that $C_1 \cup \dots \cup C_q = \{S_1, \dots, S_{p-1}, S_p, S_{p+1}\}$, with $S_p := u$ and $S_{p+1} := v$. For
 274 each $i \in [p + 1]$, S_i belongs to a unique C_j with $j \in [q]$: let $j(i) \in [q]$ be such that $S_i \in C_{j(i)}$.

275 By $\mathcal{P}(k + 1)$ and up to interchanging labels, for every $j \in [q]$ there exists a $(\kappa + 1)$ -
 276 expression φ_{C_j} of the p -labelled graph $G_{C_j} = G[\bigcup C_j]$ with for all $i \in [p]$ with $j(i) = j$,
 277 $V_{G_{C_j}}^i = S_i$. Therefore, $\varphi' := \varphi_{C_1} \oplus \dots \oplus \varphi_{C_q}$ expresses the disjoint union of the graphs
 278 G_{C_1}, \dots, G_{C_q} . Furthermore, φ' is an expression of a graph over the same vertices as $G[\bigcup C]$,
 279 however, we still need to construct the black edges crossing these red-connected components.

280 We thus apply $\eta_{i,i'}$ to φ' for every black edge of the form $(S_i, S_{i'})$ in G_{k+1} , to obtain an
 281 expression φ'' . Since the vertices with labels i and i' are exactly the vertices of S_i and $S_{i'}$,
 282 we create exactly the edges between vertices of S_i and of $S_{i'}$. By Property 1, we construct
 283 the correct black edges in $G[\bigcup C]$, and thus φ'' is an expression of $G[\bigcup C]$.

284 Moreover, we need to make sure that the labels in φ'' match the expectations of $\mathcal{P}(k)$.
 285 For that, we apply $\rho_{p+1 \rightarrow p}$ to φ'' to get an expression φ_{G_C} . By doing so, S_p (say, u) and
 286 S_{p+1} (say, v) have the same label in φ_{G_C} . Hence, φ_{G_C} witnesses $\mathcal{P}(k)$ (since $S_p = u$ and
 287 $S_{p+1} = v$ are now contracted together as the vertex $S'_p = uv$ in G_k) for the red-connected
 288 component C . Indeed, we have used $p + 1 = |C| + 1 \leq \kappa + 1$ different labels to construct
 289 φ_{G_C} from $\varphi_{C_1}, \dots, \varphi_{C_q}$.

290 Since $\{V_G\}$ is a red-connected component of $G_1 = (\{V_G\}, \emptyset, \emptyset)$, it follows from $\mathcal{P}(1)$ that
 291 $G[V_G] = G$ has a $(\kappa + 1)$ -expression, and thus $\text{cw}(G) \leq \kappa + 1$. Recall that $\kappa = \text{ctww}(G)$,
 292 and thus $\text{cw}(G) \leq \text{ctww}(G) + 1$. ◀

293 Moreover, if the contraction sequence given is linear, we obtain a linear $(\kappa + 1)$ -expression.
 294 We thus get for all graph G that $\text{linearcw}(G) \leq \text{linearctww}(G) + 1$.

295 Notice that linearcw can not be bounded by a function of ctww . For instance, the class
 296 of cographs (and even of trees) have unbounded linear clique-width [22], despite having a
 297 bounded component twin-width of 1. We now prove the rightmost bound.

298 ► **Theorem 4.** For every graph G , we have:

299 (i) $\text{ctww}(G) \leq 2\text{cw}(G) - 1$, and

300 (ii) $\text{ctww}(G) \leq \text{linearcw}(G)$.

301 **Proof.** We first prove (i) and then adapt it to prove (ii). Let $k := \text{cw}(G)$ and take a
 302 k -expression of G . We will explain how to construct a contraction sequence of G in which
 303 every red-connected component has size $\leq 2k - 1$. The following remark will be implicitly
 304 used throughout this proof.

305 ► **Remark 5.** Two vertices that have the same label in an expression φ' also have the same
 306 label in any expression of φ that has φ' as a sub-expression.

307 We prove the following property of k -expressions of φ by structural induction:
 308 $\mathcal{H}(\varphi)$: “Let $(G, l_G) := [\varphi]$. There exists a (partial) contraction sequence $(G_n, \dots, G_{k'})$ with
 309 $k' \leq k$ of G such that:

- 310 ■ every red-connected component in the trigraphs $G_n, \dots, G_{k'}$ has size $\leq 2k - 1$,
- 311 ■ the vertices of $G_{k'}$ are exactly the non-empty V_G^i for $i \in [k]$, and
- 312 ■ every pair of vertices contracted have the same labels in $(G, l_G)^4$.

313 If $\varphi = \bullet_i$ with $i \in [k]$, there is nothing to do since G has only one vertex. If φ is of the
 314 form $\rho_{i \rightarrow j}(\varphi')$ (with $(i, j) \in [k]^2$ and $i \neq j$), consider for G the partial contraction sequence
 315 of $(G', l_{G'}) := [\varphi']$ given by $\mathcal{H}(\varphi')$, and then contract $V_{G'}^i$ and $V_{G'}^j$ to obtain $V_G^j = V_{G'}^i \cup V_{G'}^j$.
 316 Since φ' is also a k -expression of G , this partial contraction sequence of G satisfies $\mathcal{H}(\varphi)$.

317 If φ is of the form $\eta_{i,j}(\varphi')$ (with $(i, j) \in [k]^2$ and $i \neq j$), consider for G the partial
 318 contraction sequence of $(G', l_{G'}) := [\varphi']$ given by $\mathcal{H}(\varphi')$. To prove that it is sufficient to prove
 319 $\mathcal{H}(\varphi)$, it is sufficient to justify that it does not create any red edge in the contraction of G
 320 that was not present in the contraction of G' . The first red-edge (x, y) that would appear in
 321 the contraction of $G = [\eta_{i,j}(\varphi')]$ that does not appear in the same contraction of $G' = [\varphi']$,
 322 results necessarily of the contraction of two vertices u and v with $x = uv$ and y being in the
 323 symmetric difference of the neighborhoods of u and v in $G = [\eta_{i,j}(\varphi')]$ but not in $G' = [\varphi']$.
 324 Such a red-edge can not exist because we contract only vertices with the same label in φ'
 325 (or, equivalently, in φ), and that $\eta_{i,j}$ can only decrease (w.r.t. \subseteq) the symmetric difference
 326 between the neighborhood of vertices with the same label in φ . By Remark 5, this implies
 327 that it is also true for vertices having the same label in any subexpression of φ .

328 If φ is of the form $\varphi = \varphi' \oplus \varphi''$: denote $(G', l') := [\varphi']$ and $(G'', l'') := [\varphi'']$, thereby,
 329 $V_G = V_{G'} \cup V_{G''}$. Consider the partial contraction sequence of G given by:

- 330 1. contract the vertices in $V_{G'}$ in accordance to the contraction sequence given by $\mathcal{H}(\varphi')$,
- 331 2. contract the vertices in $V_{G''}$ in accordance to the contraction sequence given by $\mathcal{H}(\varphi'')$
- 332 3. for all $i \in [k]$, contract $V_{G'}^i$ with $V_{G''}^i$ (if both are nonempty) to get $V_G^i = V_{G'}^i \cup V_{G''}^i$.

333 Steps 1 and 2 do not create a red-edge adjacent to both $V_{G'}$ and $V_{G''}$ (since these are
 334 two distinct connected components of G).

335 Thus, before step 3, we have a trigraph with $\leq 2k$ vertices (because both trigraphs
 336 obtained after $\mathcal{H}(\varphi')$ and $\mathcal{H}(\varphi'')$ have less than k vertices), and every red-component that
 337 have appeared so far has size $\leq 2k - 1$. After the first contraction of step 3, the resulting
 338 trigraph has $\leq 2k - 1$ vertices, and thus no red-connected component of size $> 2k - 1$ can
 339 emerge. Such a contraction satisfies every requirement of $\mathcal{H}(\varphi)$. We have thus proven $\mathcal{H}(\varphi)$
 340 for every k -expression.

⁴ Inductively, we say that the label of a vertex $S \in V_{G_l}$ ($k' \leq l \leq n$) is then the common label of the
 vertices that have been contracted together to produce S .

341 Now, take a k -expression φ of G . Up to applying $\rho_{i \rightarrow 1}$ for all $i \in [k]$ to φ , we can assume
 342 that $(G, l_G) := [\varphi]$ with l_G being constant equal to 1. The partial contraction sequence of G
 343 given by $\mathcal{H}(\varphi)$ is a total contraction sequence of G of component twin-width $\leq 2k - 1$. Since
 344 $k = \mathbf{cw}(G)$, we have proven that $\mathbf{ctww}(G) \leq 2\mathbf{cw}(G) - 1$.

345 To prove (ii), we show a similar property $\mathcal{H}_{\text{lin}}(\varphi)$ for every linear k -expression. The
 346 only difference between \mathcal{H}_{lin} and \mathcal{H} is that we replace the condition $\leq 2k - 1$ by $\leq k$. The
 347 proof then follows exactly the same steps, except for the case $\varphi = \varphi' \oplus \varphi''$, where step 2
 348 (the contraction according to $\mathcal{H}_{\text{lin}}(\varphi'')$) is not necessary anymore, since φ'' is of the form \bullet_i
 349 ($i \in [k]$), and we obtain a trigraph of size $k + 1$ instead of $2k$, since φ'' has 1 vertex instead
 350 of k . This ensures that every red-connected component has size $\leq (k + 1) - 1 = k$ instead of
 351 $2k - 1$ in the non-linear case.

352 For step 3, *i.e.*, contracting vertices of the same color in φ' and in φ'' , just note that it
 353 consists of at most 1 contraction instead of k in the linear case. ◀

354 Moreover, in the proof of (i), if the k -expression given is linear, we obtain a linear
 355 contraction sequence. We get that for every graph G , $\mathbf{linearctww}(G) \leq 2\mathbf{linearcw}(G) - 1$.

356 4 Approximating Component Twin-Width

357 The linear bounds established in Section 3 entail reasonable approximation results for
 358 component twin-width by making use of known approximations of clique-width [26]. The
 359 best currently known approximation algorithm for clique-width is given by Theorem 6.

360 ▶ **Theorem 6.** [26] *Let k be a fixed positive integer. There is an $O(|V_G|^3)$ -time algorithm that
 361 either outputs an $(8^k - 1)$ -expression of an input graph G or confirms that the clique-width
 362 of G is larger than k .*

363 From Theorem 6 and the linear bounds established in Theorem 3 and Theorem 4, we
 364 immediately obtain an approximation algorithm for component twin-width.

365 ▶ **Theorem 7.** *Let p be a fixed positive integer. There is an $O(|V_G|^3)$ -time algorithm that
 366 either outputs a contraction sequence of component twin-width $\leq 2^{3p+4} - 3$ of an input graph
 367 G or confirms that the component twin-width of G is larger than p .*

368 **Proof.** The algorithm consists of applying the algorithm of Theorem 6 to G with $k := p + 1$.
 369 If the algorithm confirms that $\mathbf{cw}(G) > p + 1$, then we know that $\mathbf{ctww}(G) > p$ by Theorem
 370 3. Otherwise, it outputs a $(2^{3(p+1)} - 1)$ -expression of G , which we transform into a contraction
 371 sequence of G of component twin-width $\leq 2 \times (2^{3(p+1)} - 1) - 1 = 2^{3p+4} - 3$ through the
 372 constructive proof of Theorem 4, which can be performed in linear time in the size of the
 373 $(2^{3(p+1)} - 1)$ -expression of G . ◀

374 It is still interesting to see that a direct comparison between component twin-width and
 375 rank-width yields to a better approximation ratio, thanks to Theorem 8. In fact, Theorem 6
 376 was also obtained by this method. By avoiding using clique-width as an intermediate
 377 parameter, it is not surprising that we obtain a better ratio.

378 ▶ **Theorem 8.** [26] *Let k be a fixed positive integer. There is an $O(|V(G)|^3)$ -time algorithm
 379 that either outputs a rank-decomposition (of an input graph G) of width at most $3k - 1$ or
 380 confirms that the rank-width of G is larger than k .*

381 We can indeed make use of the bounds given by Theorem 9. The proof is very similar to
 382 the proof of functional equivalence between boolean-width and component twin-width [8],
 383 which is not surprising, since Remark 2 applies both to rank-width and boolean-width.

384 ► **Theorem 9.** *For every graph G , $\mathbf{rw}(G) - 1 \leq \mathbf{ctww}(G) \leq 2^{\mathbf{rw}(G)+1}$*

385 **Proof.** The first inequality $\mathbf{rw}(G) - 1 \leq \mathbf{ctww}(G)$ follows from Theorem 3, stating that
386 $\mathbf{cw}(G) - 1 \leq \mathbf{ctww}(G)$ and the fact that $\mathbf{rw}(G) \leq \mathbf{cw}(G)$ [25].

387 We now focus on proving the second bound of $\mathbf{ctww}(G) \leq 2^{\mathbf{rw}(G)+1}$. This proof follows
388 the same scheme as the proof of the functional equivalence between boolean-width and
389 component twin-width [8], primarily since Remark 2 also applies to both boolean-width.

390 Similarly to a branch-decomposition of graphs, a branch-decomposition of a trigraph G'
391 is a ternary tree whose set of leaves is $V_{G'}$. It is said to be rooted if a non-leaf vertex has
392 been chosen to be the root, which leads to the usual definition of children and descendants
393 in a rooted tree. The set of leaves descending from a vertex v of a tree T is denoted by D_v .
394 Now, let G be a graph and T be an optimal branch decomposition of G , and let $r := \mathbf{rw}(G)$.
395 We prove by induction the following property for $k \in [n]$.

396 $\mathcal{P}(k)$: “There exists a (partial) contraction sequence (G_n, \dots, G_k) of G of component
397 twin-width $\leq 2^{r+1}$. Moreover, there exists a branch-decomposition T_k of G_k such that for
398 every $t \in V_{T_k}$ with $|D_t| > 2^r$, there is no red-edge crossing the bipartition $(D_t, V_{G_k} \setminus D_t)$.”

399 Note that $\mathcal{P}(n)$ is indeed true since $G = G_n$ has no red-edge. Now assume $\mathcal{P}(k+1)$ with
400 $k \in [n-1]$. We will prove $\mathcal{P}(k)$. First, note that if $k \leq 2^r$, contracting any two arbitrary
401 vertices and giving any branch decomposition of G_k proves $\mathcal{P}(k)$. We may thus assume that
402 $k > 2^r$. The root ρ satisfies $|D_\rho| \geq 2^r + 1$. Observe that there exists a node v of T_{k+1} such
403 that $2^r + 1 \leq |D_v| \leq 2^{r+1}$: a node v such that D_v has size at least $2^r + 1$ and which is
404 furthest from the root meets the condition. Moreover, any child w of v verifies $|D_w| \leq 2^r$.

405 If we use Remark 2 with respect to an edge adjacent to v , then there are two vertices u
406 and u' that satisfy $N_G(u) \cap (V_{G_{k+1}} \setminus D_v) = N_G(u') \cap (V_{G_{k+1}} \setminus D_v)$.

407 To prove $\mathcal{P}(k)$, we will prove that it is sufficient to contract the vertices u and u' of G_{k+1}
408 to obtain G_k , and to identify the leaves u and u' of T_{k+1} to obtain T_k (ie. we remove u' and
409 shortcut every degree 2 vertex that appear, and we then rename u as uu'). Note that all the
410 red-edges created by the contraction of u and u' are adjacent to the new vertex uu' .

411 First, by our choice of u and u' , we do not create any red-edge crossing $(D_v, V_{G_k} \setminus D_v)$.
412 Due to the property of T_{k+1} ensured by $\mathcal{P}(k+1)$ (recall that $|D_v| > 2^r$), there is no red-edge
413 crossing $(D_v, V_{G_k} \setminus D_v)$ in T_k . The red-connected component C of the new vertex uu' is
414 thus contained in D_v , and thus has size at most 2^{r+1} . Since C is the only red-connected
415 component of G_k that was not a red-connected component of G_{k+1} , G_k indeed meets the
416 requirements of $\mathcal{P}(k)$.

417 Second, due to the choice of v , any node t of T_k with $|D_t| > 2^r$ containing the new vertex
418 uu' is an ancestor of v . Since $D_v \subseteq D_t$, by the above argument, there is no red-edge crossing
419 $(D_t, V_{G_k} \setminus D_t)$.

420 The proof of $\mathcal{P}(k)$ is now complete: $\mathcal{P}(1)$ justifies that $\mathbf{ctww}(G) \leq 2^{r+1}$. ◀

421 This bounds naturally leads to the approximation given in Theorem 10.

422 ► **Theorem 10.** *Let p be a fixed positive integer. There is an $O(|V_G|^3)$ -time algorithm that
423 either outputs a contraction sequence of component twin-width $\leq 8^{p+1}$ of an input graph G
424 or confirms that the component twin-width of G is larger than p .*

425 **Proof.** This result can be obtained similarly to Theorem 7, by using Theorems 8 and 9
426 instead of Theorems 3 and 4 and 6. ◀

5 Complexity Results

We show two algorithmic applications of dynamic programming over component twin-width to $\#H$ -COLORING. The first assumes that an optimal contraction sequence of the input graph G is given, and results in a FPT algorithm parameterized by \mathbf{ctww} , running in time $O^*((2^{|V_H|} - 1)^{\mathbf{ctww}(G)})$. The second approach uses an optimal contraction sequence of the template H (whose computation can be seen as a pre-computation, since it does not involve the input graph G): we obtain a fine-grained algorithm running in time $O^*((\mathbf{ctww}(H) + 2)^{|V_G|})$, whose complexity beats the best algorithms of the literature and that runs in time $O^*((2\mathbf{cw}(H) + 1)^{|V_G|})$ and $O^*((\mathbf{linearcw}(H) + 2)^{|V_G|})$ [27] through the linear bounds of Section 3. Note that the technique employed in this paper could similarly be used to derive the same complexity results applied to the more general frameworks of counting the solutions of *binary constraint satisfaction problems*, ie. problems of the forms $\#\text{BINARY-CSP}(\Gamma)$ with Γ a set of binary relations over a finite domain, even though we restrict to the simpler case of $\#H$ -COLORING here to avoid having to define contraction sequences of instances and template of binary constraint satisfaction problems.

5.1 Parameterized complexity

We present an algorithm solving $\#H$ -COLORING in FPT time parameterized by component twin-width. It is inspired by the algorithm solving k -COLORING [8], thus proving that $\#H$ -COLORING is FPT parameterized by component twin-width and thus also by clique-width (by functional equivalence). The proof can be found in Appendix A.

► Theorem 11. *For any graph H , there exists an algorithm running in time $O^*((2^{|V_H|} - 1)^{\mathbf{ctww}(G)})$ that solves $\#H$ -COLORING on any input graph G (assuming that an optimal contraction sequence (G_n, \dots, G_1) of G is given).*

If one only wishes to solve H -COLORING rather than the counting problem, the algorithm by Ganian et al. [21] which runs in $O^*(s(H)^{\mathbf{cw}(G)})$ for a graph parameter s , is strictly more efficient. Indeed, for any graph H , its structural parameter $s(H)$ is bounded by $2^{|V_H|} - 2$ [21] (the equality happens if and only if H is a clique), and as we have proven in Theorem 3, for any graph G , $\mathbf{cw}(G) \leq \mathbf{ctww}(G) + 1$. However, it is difficult to extend this algorithm to the counting problem since the sets stored as invariants in the algorithm do not necessarily represent disjoint subsets of partial coloring. This is acceptable if one only wants to determine the existence of a total coloring (as long as every coloring is represented at least once), but which causes issues when counting the number of colorings.

More precisely, for the $\#k$ -COLORING problem, one may remark that the $O^*((2^k - 2)^{\mathbf{cw}(G)})$ algorithm by [24] also solves the counting problem, and thus runs faster than the algorithm given in the proof of Theorem 11. However, our algorithm is strictly more general since it is applicable to *any* $\#H$ -COLORING problem.

5.2 Fine-grained complexity

We now consider the dual problem of solving $\#H$ -COLORING when H has bounded component twin-width. We therefore use an optimal contraction sequence of the template H instead of the input G , and obtain a fine-grained algorithm for $\#H$ -COLORING which runs in $O^*((\mathbf{ctww}(H) + 2)^n)$ time. The algorithm is inspired by the algorithms running in $O^*((2\mathbf{cw}(H) + 1)^n)$ and $O^*((\mathbf{linearcw}(H) + 2)^n)$ time by Wahlström [27], but it is *always* at least as fast as both.

► **Theorem 12.** $\#H$ -COLORING is solvable in time

$$O^*((\mathbf{ctww}(H) + 2)^{|V_G|}).$$

The proof can be found in Appendix B. By Theorem 4, $\mathbf{ctww}(H) + 2 \leq \mathbf{linearcw}(H) + 2$ and $\mathbf{ctww}(H) + 2 \leq 2\mathbf{cw}(H) + 1$ for any graph H . Therefore, the algorithm in the proof of Theorem 12 is always asymptotically faster than the clique-width approach by Wahlström [27] that solves the problem in $O^*((\mathbf{linearcw}(H) + 2)^{|V_G|})$ and $O^*((2\mathbf{cw}(H) + 1)^{|V_G|})$ time.

6 Conclusion and Future Research

In this paper we explored component twin-width in the context of $\#H$ -COLORING problems. We improved the bounds of the functional equivalence between component twin-width and clique-width from the (doubly) exponential $\mathbf{cw} \leq 2^{2^{\mathbf{ctww}}}$ and $\mathbf{ctww} \leq 2^{\mathbf{cw}+1}$ to the linear

$$\mathbf{cw} \leq \mathbf{ctww} + 1 \leq 2\mathbf{cw}.$$

In particular, this entails a practical FPT algorithm for H -COLORING parameterized by component twin-width. From these linear bounds derives an approximation algorithm with exponential ratio, that can even be improved by a direct comparison with rank-width. Finally, we constructed two algorithms for solving $\#H$ -COLORING. The first uses a given optimal contraction sequence of the input graph G to solve $\#H$ -COLORING in FPT time parameterized by component twin-width. The second uses a contraction sequence of the template graph H and beats the clique-width approach for solving $\#H$ -COLORING (with respect to $|V_G|$).

We now discuss some topics for future research.

6.1 Tightness of the bounds

Even though the bound $\mathbf{cw} \leq \mathbf{ctww} + 1$ given by Theorem 3 is tight for any cograph with at least 1 edge, we do not currently know if this bound can be improved for graphs with greater clique-width or component twin-width. Moreover, it would be interesting to determine whether the bound $\mathbf{ctww} \leq 2\mathbf{cw} - 1$ given by Theorem 4 is tight. The same remark holds for the bounds between component twin-width and rank-width given by Theorem 9. It would be interesting to study the tightness of the bound $\mathbf{tww} \leq 2\mathbf{cw} - 2$ (where \mathbf{tww} designs the twin-width), which is a direct consequence of Theorem 4.

6.2 Lower bounds on complexity

The algorithms relying on clique-width to solve H -COLORING by [21] in $O^*(s(H)^{\mathbf{cw}(G)})$ time are known to be optimal under the SETH. We have a similar optimality result for treewidth (\mathbf{tw}), with an algorithm solving H -COLORING in time $|V_H|^{\mathbf{tw}(G)}$, despite the existence of an algorithm in $(|V_H| - \varepsilon)^{\mathbf{tw}(G)}$ with $\varepsilon > 0$ being ruled out under SETH. A natural research direction is then to optimize the running time of the algorithm of Theorem 11, possibly by making use of $s(H)$, and prove a similar lower bound.

6.3 Extensions

Instead of solving $\#H$ -COLORING the results of Section 5 can be extended to arbitrary binary constraints (*binary constraint satisfaction problems*, BCSPs). The notion of component twin-width indeed generalize naturally to both instances and a templates of a BCSP.

502 Additionally, one may note that the algorithms detailed in Section 5 can be adapted
503 to solve a “cost” version of $\#H$ -COLORING: given a weight matrix C , the cost of an
504 homomorphism f is $\sum_{u \in V_G} C(u, f(u))$, and we want to find an homomorphism of minimal cost.



A Proof of Theorem 11

► **Theorem 11.** *For any graph H there exists an algorithm running in time $O^*((2^{|V_H|} - 1)^{\text{ctww}(G)})$ which solves $\#H$ -COLORING on any input graph G (assuming that an optimal contraction sequence (G_n, \dots, G_1) of G is given).*

Proof. For $k \in [n]$, $C = \{S_1, \dots, S_p\} \subseteq V_{G_k}$ a red-connected component of vertices of G_k , and for $\gamma : C \mapsto (2^{V_H} \setminus \emptyset)$, a H -coloring of $G[\cup C]$ with profile γ is a H -coloring f of $G[\cup C]$ such that for all $i \in [p]$, $f(S_i) = \gamma(S_i)$. I.e. the vertices of H used to color S_i are exactly the colors of the set $\gamma(S_i)$.

Then, define the set $COL(C, \gamma)$ as the set of H -colorings of $G[\cup C]$ with profile γ . We see that for every red-connected component C of G_k , the sets $COL(C, \gamma)$ for $\gamma : C \mapsto (2^{V_H} \setminus \emptyset)$ form a partition of the set of the H -colorings of $G[\cup C]$.

The principle of the algorithm is, similarly to the proof of Theorem 2 in the main text, to inductively maintain (from $k = n$ to 1) the knowledge of every $|COL(C, \gamma)|$ (stored in a tabular $\#col(C, \gamma)$) for a red-connected component C of G_k and $\gamma : C \mapsto (2^{V_H} \setminus \emptyset)$. In this way, since $\{V_G\}$ is a red-connected component of $G_1 = (\{V_G\}, \emptyset, \emptyset)$, we can obtain the number of H -colorings of $G[V_G] = G$ by computing

$$\sum_{T \in (2^{V_H} \setminus \emptyset)} \#col(\{V_G\}, V_G \mapsto T).$$

First, note that the red-connected components of G_n are the $\{u\}$ for $u \in V_G$ (since $G_n = (V_G, E_G, \emptyset)$ has no red edge). For every $\gamma : u \mapsto \gamma(u) \in (2^{V_H} \setminus \emptyset)$ we let $\#col(\{u\}, \gamma) \leftarrow 0$ if $|\gamma(u)| \neq 1$ and $\#col(\{u\}, \gamma) \leftarrow 1$ if $|\gamma(u)| = 1$. Hence, we correctly store the value of $|COL(\{u\}, \gamma)|$ in the tabular $\#col(\{u\}, \gamma)$.

We explain how to maintain this invariant after the contraction from G_{k+1} to G_k (with $k \in [n - 1]$). By definition of a contraction sequence, G_k is of the form $G_k =: G_{k+1}/(u, v)$ with u and v two different vertices of G_{k+1} .

Note that every red-connected component of G_k is also a red-connected component of G_{k+1} , except the red-connected component C containing uv . We only have to compute $|COL(C, \gamma)|$ for any $\gamma : C \mapsto 2^{V_H} \setminus \emptyset$, and to store it in the tabular $\#col(C, \gamma)$. Initialize the value of $\#col(C, \gamma)$ with 0.

Let $C =: \{S_1, \dots, S_{p-1}, S'_p\}$, with $S'_p := uv$, and $p := |C| \leq \text{ctww}(G)$. Since every pair of red-connected vertices in G_{k+1} (that contains neither u nor v) are red-connected in G_k (except u and v), C must be of the form

$$C := (C_1 \cup \dots \cup C_q \cup \{S'_p\}) \setminus \{S_p, S_{p+1}\},$$

with $S_p := u$ and $S_{p+1} := v$ and $C_1 \cup \dots \cup C_q = \{S_1, \dots, S_{p-1}, S_p, S_{p+1}\}$,⁵ and where C_1, \dots, C_q (with $q > 0$) are red-connected components of G_{k+1} whose union contains both $S_p = u$ and $S_{p+1} = v$. Notice that each S_i (for $i \in [p + 1]$) belongs to a unique $C_{j(i)}$ with $j(i) \in [q]$.

An illustration of these notations are given in Figure 2.

The algorithm iterates over every family $(\gamma_j : C_j \mapsto (2^{V_H} \setminus \emptyset))_{1 \leq j \leq q}$. Let $\gamma = \gamma_1 \cup \dots \cup \gamma_q$ that maps every S_i (with $i \in [p - 1]$) to $\gamma_{j(i)}(S_i)$, and that maps $S'_p = uv = S_p \cup S_{p+1} = u \cup v$ to $\gamma_{j(p)}(S_p) \cup \gamma_{j(p+1)}(S_{p+1})$. We check if there exists a $(i, i') \in [p]^2$ with $i \neq i'$, a black edge

⁵ Note that $uv = S'_p = S_p \cup S_{p+1} = u \cup v$.

535 between S_i and $S_{i'}$ in G_{k+1} , and $(\gamma(S_i) \times \gamma(S_{i'})) \setminus E_H \neq \emptyset$, in time $O(p^2)$. If so, we move to
 536 the next family $(\gamma_j)_{1 \leq j \leq q}$. Otherwise, we increment $\#col(C, \gamma)$ by $\prod_{j=1}^q \#col(C_j, \gamma_j)$.

537 **Soundness:** The soundness of this algorithm follows from the fact that for each $\gamma : C \mapsto$
 538 $2^{V_H} \setminus \emptyset$, $COL(C, \gamma)$ is the disjointed union, for $(\gamma_1, \dots, \gamma_q)$ such that $\gamma = \gamma_1 \cup \dots \cup \gamma_q$, of
 539 the sets of H -colorings f such that for all $j \in [q]$ the profile of $f|_{C_j}$ is γ_j , that we denote by
 540 $COL(C, \gamma_1, \dots, \gamma_q)$. We only need to compute $|COL(C, \gamma_1, \dots, \gamma_q)|$, which can be derived by
 541 Claim 13. We then store the sum over $(\gamma_1, \dots, \gamma_q)$ such that $\gamma = \gamma_1 \cup \dots \cup \gamma_q$ in $\#col(C, \gamma)$.

542 \triangleright **Claim 13.** There are two distinct cases:

- 543 1. If there exists $(i, i') \in [p]^2$ such that $(S_i, S_{i'})$ is a black edge of G_{k+1} and $(\gamma_{j(i)}(S_i) \times$
 544 $\gamma_{j(i')}(S_{i'})) \setminus E_H \neq \emptyset$, then $COL(C, \gamma_1, \dots, \gamma_q) = \emptyset$.
- 545 2. If for all $(i, i') \in [p]^2$ such that $(S_i, S_{i'})$ is a black edge of G_{k+1} , $(\gamma_{j(i)}(S_i) \times \gamma_{j(i')}(S_{i'})) \subseteq$
 546 E_H , then a function $f : \cup C \mapsto V_H$ belongs to $COL(C, \gamma_1, \dots, \gamma_q)$ iff, for all $j \in [q]$, f
 547 restricted to C_j (denoted by f_j) belongs to $COL(C_j, \gamma_j)$.

548 **Proof.** We treat the two cases separately. In the first case, assume that there exists
 549 $(i, i') \in [p]^2$ such that $(S_i, S_{i'})$ is a black edge of G_{k+1} and $(\gamma_{j(i)}(S_i) \times \gamma_{j(i')}(S_{i'})) \setminus E_H \neq \emptyset$
 550 and, for the sake of contradiction, suppose that there is $f \in COL(C, \gamma_1, \dots, \gamma_q)$. Take
 551 $(v_i, v_{i'}) \in (\gamma_{j(i)}(S_i) \times \gamma_{j(i')}(S_{i'})) \setminus E_H$. By definition of a profile, there exists $(u_i, u_{i'}) \in S_i \times S_{i'}$
 552 with $f(u_i) = v_i$ and $f(u_{i'}) = v_{i'}$. Then, since there exists a black edge between S_i and $S_{i'}$ in
 553 G_{k+1} , this means by Property 1 that $(u_i, u_{i'}) \in E_G$. But $(f(u_i), f(u_{i'})) = (v_i, v_{i'}) \notin E_H$, so
 554 f is not a H -coloring, which contradicts the definition of f .

555 In the second case, assume that for all $(i, i') \in [p]^2$ such that $(S_i, S_{i'})$ is a black edge
 556 of G_{k+1} , $(\gamma_{j(i)}(S_i) \times \gamma_{j(i')}(S_{i'})) \subseteq E_H$. To prove necessity, notice that the restriction of a
 557 partial H -coloring is also a partial H -coloring, and by definition of $COL(C, \gamma_1, \dots, \gamma_q)$, if
 558 $f \in COL(C, \gamma_1, \dots, \gamma_q)$, then $f_j \in COL(C_j, \gamma_j)$.

559 To prove sufficiency, assume that $f : \cup C \mapsto V_H$ is such that for all $j \in [q]$, $f_j \in$
 560 $COL(C_j, \gamma_j)$. Then, provided that f is a H -coloring of $G[\cup C]$, $f \in COL(C, \gamma_1, \dots, \gamma_q)$.
 561 Hence, we only have to prove that f is a H -coloring. So let $(u, u') \in E_G$. We prove that
 562 $(f(u), f(u')) \in E_H$. Observe that there exist S_i and $S_{i'}$ (with $(i, i') \in [p]^2$) such that $u \in S_i$
 563 and $u' \in S_{i'}$. If S_i and $S_{i'}$ are in the same red-connected component C_j (with $j \in [q]$)
 564 of G_{k+1} , then $(f(u), f(u')) = (f_j(u), f_j(u')) \in E_H$ because f_j is a H -coloring. Otherwise,
 565 $(S_i, S_{i'})$ is not a red edge of G_{k+1} , so $(S_i, S_{i'})$ is a black edge of G_{k+1} , since $(u, u') \in E_G$
 566 and $(u, u') \in S_i \times S_{i'}$, by Property 1. By assumption, $(\gamma_{j(i)}(S_i) \times \gamma_{j(i')}(S_{i'})) \subseteq E_H$ and, by
 567 definition of a profile, $(f(u), f(u')) = (f_{j(i)}(u), f_{j(i')}(u')) \in \gamma_{j(i)}(S_i) \times \gamma_{j(i')}(S_{i'}) \subseteq E_H$. The
 568 latter shows that f is indeed a H -coloring. \blacktriangleleft

569 From Claim 13 it follows that choosing an f in $COL(C, \gamma_1, \dots, \gamma_q)$ is either impossible,
 570 or equivalent to choosing $f_j \in COL(C_j, \gamma_j)$ for all $j \in [q]$, which is why we add either 0 or
 571 $\prod_{j=1}^q \#col(C_j, \gamma_j)$ when treating the part of $\#col(C, \gamma)$ relative to the family $(\gamma_1, \dots, \gamma_q)$.

Complexity: To treat the red-connected component C , the only non-polynomial part is
 to iterate over every family $(\gamma_1, \dots, \gamma_q)$, which represents

$$\prod_{j=1}^q (2^{|V_H|} - 1)^{|C_j|} = (2^{|V_H|} - 1)^{|C|+1} \leq (2^{|V_H|} - 1)^{\text{ctww}(G)+1}$$

572 families to treat (recall that for all $j \in [q]$, γ_j is a non-empty subset of C_j). \blacktriangleleft

B Proof of Theorem 12

► **Theorem 12.** *#H-COLORING is solvable in time*

$$O^*((\mathbf{ctww}(H) + 2)^{|V_G|}).$$

Proof. Consider an optimal contraction sequence (H_m, \dots, H_1) of H , with $m := |V_H|$. We give an algorithm similar to that described in the proof of Theorem 11, except that we define profiles for red-connected component of each H_k , $k \in [m]$.

Let $C = \{T_1, \dots, T_p\}$ be a red connected component of H_k and let $\gamma = (S_1, \dots, S_p)$ be a p -tuple of pairwise disjoint subsets of V_G . An H -coloring f of $G[S_1 \cup \dots \cup S_p]$ is said to have C -profile γ if for each $i \in [p]$, $f(S_i) \subseteq T_i$. Denote by $COL(\gamma, C)$ the set of partial H -colorings of G (i.e., a H -COLORING of an induced subgraph) with C -profile γ . It is easy to compute the $|COL(\gamma, C)|$ for a red-connected component C of $H_m = (V_H, E_H, \emptyset)$ and $\gamma = (S)$ with $S \subseteq V_G$, since C is of the form $C = \{v\}$ with $v \in E_H$. We have $|COL((S), \{v\})| = 1$ if $S^2 \cap E_G = \emptyset$, and $|COL((S), \{v\})| = 0$, otherwise.

As in the proof of Theorem 11, for $k \in [m - 1]$ the only red-connected component of H_k that is not a red-connected component of H_{k+1} , is the red-connected component $C = \{T_1, \dots, T_{p-1}, T'_p\}$ that contains $T'_p = uv$ (the vertex obtained by contraction of $T_p = u$ and $T_{p+1} = v$ in H_{k+1}). Hence, C is of the form

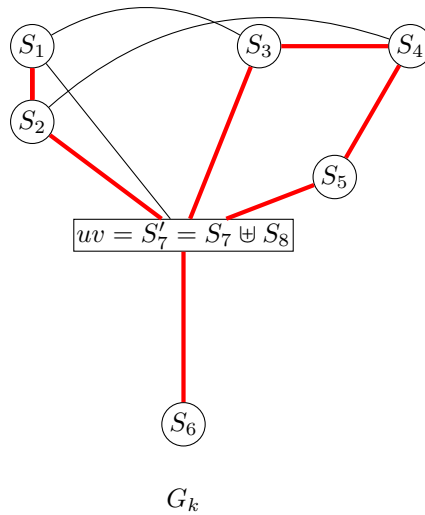
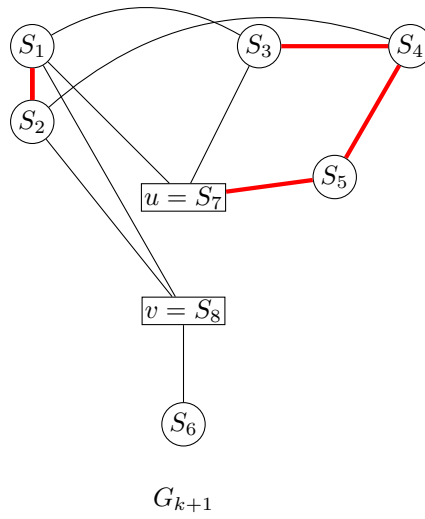
$$C = (C_1 \cup \dots \cup C_q \cup \{T'_p\}) \setminus \{T_p, T_{p+1}\},$$

with $C_1 \cup \dots \cup C_q = \{T_1, \dots, T_{p-1}, T_p, T_{p+1}\}$, where C_1, \dots, C_q are the red-connected components of H_{k+1} whose union contains $T_p = u$ and $T_{p+1} = v$. Again, each T_i belongs to a unique $C_{j(i)}$ with $j(i) \in [q]$.

Then, as in the proof of Theorem 11, for all families of disjoint subsets of V_G and $\gamma = (S_1, \dots, S_{p-1}, S'_p)$, we can compute the value of $|COL(\gamma, C)|$. Indeed, as in the proof of Theorem 11, it is the sum for every family $(\gamma_j)_{1 \leq j \leq q}$ that defines the profile γ (i.e., every γ_j is a family of pairwise disjoint subsets of V_G , and S'_p is of the form $S'_p = S_p \cup S_{p+1}$ with $S_p \cap S_{p+1} = \emptyset$ and $\forall j \in [q]$, $\gamma_j = (S_i)_{i \in j^{-1}(\{j\})}$)

of the value (1) $\prod_{j=1}^q |COL(\gamma_j, C_j)|$ if for every $(i, i') \in [p]^2$ with $j(i) \neq j(i')$ and for every edge $(u_i, u_{i'})$ of G with $u_i \in S_i$ and $u_{i'} \in S_{i'}$, there is a black edge between T_i and $T_{i'}$ in H_{k+1} , and (2) 0, otherwise.

The complexity of computing $|COL(\gamma, C)|$ for every γ is $(\mathbf{ctww}(H) + 2)^{|V_G|}$, since exploring every family $(\gamma_j)_{1 \leq j \leq q}$ containing only pairwise disjoint subsets of $|V_G|$ requires to explore $(\sum_{j=1}^q |C_j| + 1)^{|V_G|}$ families (any vertex of G can be mapped to a unique element in $\{T_1, T_2, \dots, T_{p+1}\}$ or none of them), which makes $(|C| + 2)^n \leq (\mathbf{ctww}(H) + 2)^n$ possibilities. Since $H_1 = (\{V_H\}, \emptyset, \emptyset)$, we obtain the number of such H -colorings of G in time $O^*((\mathbf{ctww}(H) + 2)^{|V_G|})$, and it is equal to $|COL(\{V_G\}, \{V_H\})|$. ◀



■ **Figure 2** An example where merging $u = S_7$ and $v = S_8$ causes $j = 4$ different red-connected components to merge into a red-connected component of size $p = 7$. With the notations of this proof, we could have $C_1 = \{S_1, S_2\}$, $C_2 = \{S_3, S_4, S_5, S_7\}$, $C_3 = \{S_6\}$ and $C_4 = \{S_8\}$. For instance, $j(1) = j(2) = 1$, $j(3) = j(4) = j(5) = j(7) = 2$, $j(6) = 3$ and $j(8) = 4$

601 ——— References ———

- 602 1 Pierre Bergé, Édouard Bonnet, and Hugues Déprés. Deciding twin-width at most 4 is
 603 NP-complete. In Mikolaj Bojanczyk, Emanuela Merelli, and David P. Woodruff, editors,
 604 *Proceedings of the 49th International Colloquium on Automata, Languages, and Programming*
 605 *(ICALP-2022)*, volume 229 of *LIPICs*, pages 18:1–18:20. Schloss Dagstuhl - Leibniz-Zentrum
 606 für Informatik, 2022. doi:10.4230/LIPICs.ICALP.2022.18.
- 607 2 Édouard Bonnet, Dibyayan Chakraborty, Eun Jung Kim, Noleen Köhler, Raul Lopes, and
 608 Stéphan Thomassé. Twin-width VIII: delineation and win-wins. *CoRR*, abs/2204.00722, 2022.

- 609 3 Édouard Bonnet and Hugues Déprés. Twin-width can be exponential in treewidth. *CoRR*,
610 abs/2204.07670, 2022. URL: <https://arxiv.org/abs/2204.07670>.
- 611 4 Édouard Bonnet, Colin Geniet, Eun Jung Kim, Stéphan Thomassé, and Rémi Watrigant.
612 Twin-width II: small classes. In *Proceedings of the 32nd ACM-SIAM Symposium on Discrete*
613 *Algorithms (SODA-2021)*, pages 1977–1996. SIAM, 2021.
- 614 5 Édouard Bonnet, Colin Geniet, Eun Jung Kim, Stéphan Thomassé, and Rémi Watrigant.
615 Twin-width III: Max Independent Set, Min Dominating Set, and Coloring. In *Proceedings of*
616 *the 48th International Colloquium on Automata, Languages, and Programming (ICALP-2021)*,
617 volume 198 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 35:1–35:20,
618 Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. URL: <https://drops.dagstuhl.de/opus/volltexte/2021/14104>, doi:10.4230/LIPIcs.ICALP.2021.35.
- 620 6 Édouard Bonnet, Colin Geniet, Romain Tessera, and Stéphan Thomassé. Twin-width VII:
621 groups. *CoRR*, abs/2204.12330, 2022.
- 622 7 Édouard Bonnet, Ugo Giocanti, Patrice Ossona de Mendez, Pierre Simon, Stéphan Thomassé,
623 and Szymon Toruńczyk. Twin-width IV: ordered graphs and matrices. In *Proceedings of*
624 *the 54th Annual ACM SIGACT Symposium on Theory of Computing (STOC-2022)*, pages
625 924–937, 2022.
- 626 8 Édouard Bonnet, Eun Jung Kim, Amadeus Reinald, and Stéphan Thomassé. Twin-width VI:
627 the lens of contraction sequences. In *Proceedings of the 2022 Annual ACM-SIAM Symposium*
628 *on Discrete Algorithms (SODA-2022)*, pages 1036–1056. SIAM, 2022.
- 629 9 Édouard Bonnet, Eun Jung Kim, Amadeus Reinald, Stéphan Thomassé, and Rémi Watrigant.
630 Twin-width and polynomial kernels. *Algorithmica*, 84:1–38, 2022.
- 631 10 Édouard Bonnet, Eun Jung Kim, Stéphan Thomassé, and Rémi Watrigant. Twin-width I:
632 tractable FO model checking. *ACM Journal of the ACM (JACM)*, 69(1):1–46, 2021.
- 633 11 Édouard Bonnet, O-joung Kwon, David R Wood, et al. Reduced bandwidth: a qualit-
634 ative strengthening of twin-width in minor-closed classes (and beyond). *arXiv preprint*
635 *arXiv:2202.11858*, "", 2022.
- 636 12 Édouard Bonnet, Jaroslav Nešetřil, Patrice Ossona de Mendez, Sebastian Siebertz, and Stéphan
637 Thomassé. Twin-width and permutations. *CoRR*, abs/2102.06880, 2021.
- 638 13 Binh-Minh Bui-Xuan, Jan Arne Telle, and Martin Vatshelle. Boolean-width of graphs.
639 *Theoretical Computer Science*, 412(39):5187–5204, 2011.
- 640 14 Bruno Courcelle and Stephan Olariu. Upper bounds to the clique width of graphs. *Discrete*
641 *Applied Mathematics*, 101(1-3):77–114, 2000.
- 642 15 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Daniel Marx, Marcin
643 Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer Publishing
644 Company, Incorporated, 1st edition, 2015.
- 645 16 Ronald de Haan and Stefan Szeider. Parameterized complexity classes beyond para- np . *Journal*
646 *of Computer and System Sciences*, 87:16–57, 2017.
- 647 17 Martin Dyer and Catherine Greenhill. The complexity of counting graph homomorphisms.
648 *Random Structures & Algorithms*, 17(3-4):260–289, 2000.
- 649 18 Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Texts in Theoretical
650 Computer Science. An EATCS Series. Springer, 2006. doi:10.1007/3-540-29953-X.
- 651 19 Fedor V. Fomin, Alexander Golovnev, Alexander S. Kulikov, and Ivan Mihajlin. Lower bounds
652 for the graph homomorphism problem. In *Proceedings of the 42nd International Colloquium*
653 *on Automata, Languages, and Programming (ICALP-2015), 2015, Kyoto, Japan, July 6-10,*
654 *2015, Proceedings, Part I*, volume 9134 of *Lecture Notes in Computer Science*, pages 481–493.
655 Springer, 2015. doi:10.1007/978-3-662-47672-7_39.
- 656 20 Piotr Formanowicz and Krzysztof Tanaś. A survey of graph coloring-its types, methods and
657 applications. *Foundations of Computing and Decision Sciences*, 37(3):223–238, 2012.
- 658 21 Robert Ganian, Thekla Hamm, Viktoriia Korchemna, Karolina Okrasa, and Kirill Simonov.
659 The fine-grained complexity of graph homomorphism parameterized by clique-width. *CoRR*,
660 abs/2210.06845, 2022.

- 661 22 Frank Gurski and Egon Wanke. On the relationship between nlc-width and linear nlc-width.
662 *Theoretical Computer Science*, 347(1-2):76–89, 2005.
- 663 23 Petr Hliněný. Twin-width of planar graphs is at most 11. *CoRR*, abs/2205.05378, 2022.
- 664 24 Michael Lampis. Finer tight bounds for coloring on clique-width. *SIAM Journal on Discrete*
665 *Mathematics*, 34(3):1538–1558, 2020.
- 666 25 Sang-il Oum. *Graphs of bounded rank-width*. Princeton University, 2005.
- 667 26 Sang-il Oum. Approximating rank-width and clique-width quickly. *ACM Transactions on*
668 *Algorithms (TALG)*, 5(1):1–20, 2008.
- 669 27 Magnus Wahlström. New plain-exponential time classes for graph homomorphism. *Theory of*
670 *Computing Systems*, 49(2):273–282, 2011.