Complexity of C_k-Coloring in Hereditary Classes of Graphs

Maria Chudnovsky
Princeton University, Princeton, NJ 08544, USA
mchudnov@math.princeton.edu

Shenwei Huang[1]
College of Computer Science, Nankai University, Tianjin 300350, China
shenweihuang@nankai.edu.cn

Paweł Rzążewski
Faculty of Mathematics and Information Science, Warsaw University of Technology, Warsaw, Poland
p.rzazewski@mini.pw.edu.pl

Sophie Spirkl
Rutgers University, Piscataway, NJ 08854, USA
sophiespirkl@gmail.com

Mingxian Zhong
Lehman College, CUNY, Bronx, NY 10468, USA
mingxian.zhong@lehman.cuny.edu

Abstract

For a graph F, a graph G is *F-free* if it does not contain an induced subgraph isomorphic to F. For two graphs G and H, an *H-coloring* of G is a mapping $f : V(G) \to V(H)$ such that for every edge $uv \in E(G)$ it holds that $f(u)f(v) \in E(H)$. We are interested in the complexity of the problem H-Coloring, which asks for the existence of an H-coloring of an input graph G. In particular, we consider H-Coloring of F-free graphs, where F is a fixed graph and H is an odd cycle of length at least 5. This problem is closely related to the well known open problem of determining the complexity of 3-Coloring of P_t-free graphs.

We show that for every odd $k \geq 5$ the C_k-Coloring problem, even in the precoloring-extension variant, can be solved in polynomial time in P_9-free graphs. On the other hand, we prove that the extension version of C_k-Coloring is NP-complete for F-free graphs whenever some component of F is not a subgraph of a subdivided claw.

2012 ACM Subject Classification Mathematics of computing → Graph algorithms

Keywords and phrases homomorphism, hereditary class, computational complexity, forbidden induced subgraph

Digital Object Identifier 10.4230/LIPIcs.ESA.2019.31

Funding *Maria Chudnovsky*: Supported by NSF grant DMS-1763817. This material is based upon work supported in part by the U. S. Army Research Laboratory and the U. S. Army Research Office under grant number W911NF-16-1-0404.
Shenwei Huang: This research is supported by the National Natural Science Foundation of China (11801284).
Sophie Spirkl: This material is based upon work supported by the National Science Foundation under Award No. DMS1802201.

Acknowledgements We are grateful to anonymous reviewers for their comments that helped improve the presentation of the paper.

[1] Corresponding author

27th Annual European Symposium on Algorithms (ESA 2019).
Editors: Michael A. Bender, Ola Svensson, and Grzegorz Herman; Article No. 31; pp. 31:1–31:15
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

For graphs G and H, a *homomorphism from G to H* is a mapping $f : V(G) \to V(H)$ such that $f(u)f(v) \in E(H)$ for every edge $uv \in E(G)$. It is straightforward to see that if H is a complete graph with k vertices, then every homomorphism to H is in fact a k-coloring of G (and vice versa). This shows that homomorphisms can be seen as a generalization of graph colorings. Because of that, a homomorphism to H is often called an *H-coloring*, and vertices of H are called *colors*. We also say that G is *H-colorable* if G has an H-coloring.

In what follows, the target graph H is always fixed. We are interested in the complexity of the H-Coloring problem, which asks whether the input graph G has an H-coloring.

1.1 Complexity of variants of H-Coloring

Since H-Coloring is a generalization of k-Coloring, it is natural to try to extend results for k-Coloring to target graphs H which are not complete graphs. For example, it is well-known that k-Coloring enjoys a complexity dichotomy: it is polynomial-time solvable if $k \leq 2$, and NP-complete otherwise. The complexity dichotomy for H-Coloring was described by Hell and Nešetřil in their seminal paper [23]: they proved that the problem is polynomial-time solvable if H is bipartite, and NP-complete otherwise.

Since then, there have been numerous studies on variants of H-Coloring. Our main focus will be on the H-Precoloring Extension problem, in which we are given a triple (G, W, h), where G is a graph, W is a subset of $V(G)$, and h is a mapping from W to $V(H)$. The problem is to decide if h can be extended to an H-coloring of G, that is, if there is an H-coloring f of G such that $f|W = h$.

Note that this problem is closely related to the List H-Coloring problem, where the input consists of a graph G with an *H-list assignment*, which is a function $L : V(G) \to 2^{V(H)}$ that assigns a subset of $V(H)$ (called *list*) to each vertex of G. We ask if there is an H-coloring f of G such that $f(v) \in L(v)$ for each $v \in V(G)$. In such a case we say that (G, L) is *H-colorable* and f is an *H-coloring of* (G, L). Clearly H-Precoloring Extension can be seen as a restriction of List H-Coloring, in which every list is either a singleton, or contains all vertices of H. This is the reason why it is sometimes called *one-or-all list homomorphism (coloring) problem* [13].

In general, variants of H-Coloring can be seen in a wider context of Constraint Satisfaction Problems (CSP). A full complexity dichotomy for this family of problems has been a long-standing open question, known as the *CSP dichotomy conjecture* of Feder and Vardi [15]. After a long series of partial results, the problem was finally solved very recently, independently by Bulatov [5] and by Zhuk [33].

A natural approach in dealing with computationally hard problems is to consider restricted instances, in hope to understand the boundary between easy and hard cases. For example, it is known that H-Coloring can be solved in polynomial time for perfect graphs, because it suffices to test whether $\omega(G) > \omega(H)$, which can be done in $O(|V(G)|^{|V(H)|})$ time. If $\omega(G) > \omega(H)$, then the answer is no, as there is no way to map the largest clique of G to H. Otherwise the answer is yes, since $\omega(G)$-coloring of G can be translated to a homomorphism of G to the largest clique of H, and thus to H. The situation changes when we consider the more general setting of H-Precoloring Extension and List H-Coloring. For any fixed graph H, List H-Coloring (and thus H-Precoloring Extension and H-Coloring) can be solved in polynomial time for input graphs with bounded tree-width. Combining this with an observation that any graph with a clique larger than $\omega(H)$ has no H-coloring, we obtain polynomial-time algorithms for chordal graphs [14]. For permutations graphs, List

H-Coloring can also be solved in polynomial time via a recursive branching algorithm [11]. For bipartite input graphs, however, 3-Precoloring Extension (i.e., K_3-Precoloring Extension) is already NP-complete [29]. Other restricted inputs have been studied too, e.g. bounded-degree graphs [16]. For more results on graph homomorphisms, we refer to the monograph by Hell and Nešetřil [22].

1.2 Graphs with forbidden induced subgraphs

A rich family of restricted graph classes comes from forbidding some small substructures. For graphs G and F, we say that G *contains* F if F is an induced subgraph of G. By *F-free graphs* we mean the class of graphs that do not contain F. Note that this class is *hereditary*, that is, it is closed under taking induced subgraphs.

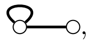
The complexity of k-Coloring for hereditary graph classes has received much attention in the past two decades and significant progress has been made. Of particular interest is the class of F-free graphs for a fixed graph F. For any fixed $k \geq 3$, the k-Coloring problem remains NP-complete for F-free graphs whenever F is not a linear forest (a collection of disjoint paths) [25, 31]. The simplest linear forests are paths, and the complexity of k-Coloring in P_t-free graphs has been studied by many researchers.

On the positive side, Hoàng, Kamiński, Lozin, Sawada, and Shu [24] gave a recursive algorithm showing that k-Coloring can be solved in polynomial time for P_5-free graphs for any fixed k. Bonomo, Chudnovsky, Maceli, Schaudt, Stein, and Zhong [4] showed that 3-Coloring can be solved in polynomial time in P_7-free graphs. Moreover, very recently, Chudnovsky, Spirkl, and Zhong proved that 4-Coloring is polynomial-time solvable in P_6-free graphs [7, 8, 9].

On the negative side, Woeginger and Sgall [32] demonstrated the NP-completeness of 5-Coloring for P_8-free graphs and 4-Coloring for P_{12}-free graphs. Later on, these NP-completeness results were improved by various researchers and the strongest result is due to Huang [26] who proved that 4-Coloring is NP-complete for P_7-free graphs and 5-Coloring is NP-complete for P_6-free graphs. These results settle the complexity of k-Coloring for P_t-free graphs for all pairs (k, t), except for the complexity of 3-Coloring for P_t-free graphs when $t \geq 8$. Interestingly, all polynomial-time results carry over to the list variant, except for the case of List 4-Coloring of P_6-free graphs, which was shown to be NP-complete by Golovach, Paulusma, and Song [18]. We refer the reader to the survey by Golovach, Johnson, Paulusma, and Song [17] for more information about coloring graphs with forbidden subgraphs.

Understanding the complexity of 3-Coloring in P_t-free graphs seems a hard problem – on the one hand, algorithms even for small values of t are difficult to construct, and on the other hand all our hardness reductions appear to introduce long induced paths. Let us mention a problem whose complexity is equally elusive: Independent Set. Alekseev [1] observed that Independent Set is NP-complete in F-free graphs whenever F is not a path or a subdivided claw. For P_t-free graphs, polynomial-time algorithms are known only for small values of t: currently, the best result is the recent polynomial-time algorithm for P_6-free graphs by Grzesik, Klimošova, Pilipczuk, and Pilipczuk [20, 21]. On the other hand, the problem is not known to be NP-hard for any fixed t.

A natural question to ask is if the similar behavior of 3-Coloring and Independent Set in P_t-free graphs is a part of a more general phenomenon. Recently, Groenland, Okrasa, Rzążewski, Scott, Seymour, and Spirkl [19] shed some light on this question by showing that if H does not contain two vertices with two common neighbors, then a very general, weighted variant of H-Coloring can be solved in time $2^{O(\sqrt{tn \log n})}$ for P_t-free graphs. Clearly K_3

does not have two vertices with two common neighbors. Moreover, Independent Set can be expressed as a weighted homomorphism to , which has the same property, and thus, for every t, both 3-Coloring and Independent Set can be solved in subexponential time in P_t-free graphs (we note that a subexponential algorithm for Independent Set in P_t-free graphs was known before [2]). This implies that if one attempts to prove NP-completeness of any of these problems in P_t-free graphs, then, assuming the Exponential Time Hypothesis [27, 28], such a reduction should be sufficiently complicated to introduce at least a quadratic blow-up of the instance.

In this paper, we study the complexity of variants of H-Coloring when H is an odd cycle of length at least five. Note that by the result of Groenland *et al.* [19], this problem can be solved in subexponential time in P_t-free graphs. We are interested in better classification of polynomial and NP-hard cases.

1.3 Our contribution

The contribution of the paper is twofold: First, we show that the C_k-Precoloring Extension problem can be solved in polynomial time in P_9-free graphs.

▶ **Theorem 1.1.** *Let $k \geq 5$ be odd, G be a P_9-free graph of order n, W be a subset of its vertices, and h be a mapping from W to $V(C_k)$. Then one can determine in $O(n^{12k+3})$ time if h can be extended to a C_k-coloring of G, and find such a C_k-coloring if one exists.*

The algorithm is described in detail in Section 3. It builds on the recent work on 3-Coloring P_7-free graphs [4]. The high-level idea of the algorithm is the following: First, we partition the graph into a so-called *layer structure* and guess the colors of a constant number of vertices. This precoloring propagates to other vertices using the layer structure, reducing the lists of possible colors. We keep guessing the colors of other vertices, transforming the input instance into a set of $n^{O(k)}$ subinstances of List H-Coloring, such that:

(i) (G, W, f) is a yes-instance of C_k-Precoloring Extension if and only if one of these subinstances is a yes-instance of List C_k-Coloring; and

(ii) each subinstance can be solved in polynomial time by a reduction to 2-Sat.

In Section 4, we study the complexity of variants of H-Coloring in F-free graphs and prove the following theorem.

▶ **Theorem 1.2.** *Let F be a connected graph. If F is not a subgraph of a subdivided claw, then for every odd $k \geq 5$ the C_k-*Precoloring Extension *problem is NP-complete for F-free graphs.*

We prove the theorem in several steps, analyzing the possible structure of F and trimming the hard cases. Observe that the statement of Theorem 1.2 is similar to the previously mentioned result of Alekseev for Independent Set [1]. In most cases, we actually prove hardness for the more restricted C_k-Coloring problem.

Finally, in Section 5, we state some open questions for future research.

2 Preliminaries

Let G be a simple graph. For $X \subseteq V(G)$, we denote by $G|X$ the subgraph induced by X, and denote by $G \setminus X$ the graph $G|(V(G) \setminus X)$. We say that X is *connected* if $G|X$ is connected. For two disjoint subsets $A, B \subset V(G)$, we say that A is *complete* to B if every vertex of A is adjacent to every vertex of B, and that A is *anticomplete* to B if every vertex of A is nonadjacent to every vertex of B. If $A = \{a\}$ we write a is complete (or anticomplete) to B to mean that $\{a\}$ is complete (or anticomplete) to B. For $X \subseteq V(G)$, we say that $e \in E(G)$ is *an edge of* X if both endpoints of e are in X. For $v \in V(G)$ we write $N_G(v)$ (or $N(v)$ when there is no danger of confusion) to mean the set of vertices of G that are adjacent to v. Observe that since G is simple, $v \notin N(v)$. For $X \subseteq V(G)$ we define $N(X) = \left(\bigcup_{v \in X} N(v)\right) \setminus X$. We say that the set S *dominates* X, or S is a *dominating set* of X if $X \subseteq S \cup N(S)$. We write that S dominates G when we mean that it dominates $V(G)$. A component of G is *trivial* if it has only one vertex and *nontrivial* otherwise.

We use $[k]$ to denote the set $\{1, 2, \ldots, k\}$. We denote by P_t the path with t vertices. A *path in a graph* G is a sequence $v_1 - \cdots - v_t$ of pairwise distinct vertices such that for any $i, j \in [t]$, $v_i v_j \in E(G)$ if and only if $|i - j| = 1$. The *length* of this path is t. We denote by $V(P)$ the set $\{v_1, \ldots, v_t\}$. If $a, b \in V(P)$, say $a = v_i$ and $b = v_j$ with $i < j$, then $a - P - b$ is the path $v_i - v_{i+1} - \cdots - v_j$, and $b - P - a$ is the path $v_j - v_{j-1} - \cdots - v_i$.

Let $k \geq 3$ be an odd integer. We denote by C_k a cycle with k vertices $1, 2, \ldots, k$ that appear along the cycle in this order. The calculations on vertices of C_k will be preformed modulo k, with 0 unified with vertex k.

We say that (G, L') is a *subinstance* of (G, L) if $L'(v) \subseteq L(v)$ for every $v \in V(G)$. Two C_k-list assignments L and L' of G are *equivalent* if (G, L) is C_k-colorable if and only if (G, L') is C_k-colorable. A C_k-list assignment L is *equivalent* to a set $\mathcal{L}$ of C_k-list assignments of a graph G if there is $L' \in \mathcal{L}$ such that (G, L) is equivalent to (G, L').

Let (G, L) be an instance of List C_k-Coloring. We say that the list $L(x)$ of a vertex x is *good* if $|L(x)| \in \{1, 2, 3, k\}$ and in addition

- if $|L(x)| = 2$, then $L(x) = \{i - 1, i + 1\}$ for some $i \in [k]$, and
- if $|L(x)| = 3$, then $L(x) = \{i, i - 2, i + 2\}$ for some $i \in [k]$.

We say that L is *good* if $L(v)$ is good for all $v \in V(G)$.

For an edge $vw \in E(G)$, we *update* v *from* w if one of the following is performed.

- If $L(w) = \{i\}$ for some $i \in [k]$, then replace the list of v by $\{i - 1, i + 1\} \cap L(v)$.
- If $L(w) = \{i-1, i+1\}$ for some $i \in [k]$, then replace the list of v by $\{i, i+2, i-2\} \cap L(v)$.
- If $L(w) = \{i, i - 2, i + 2\}$, $L(v) = \{j, j + 2, j - 2\}$ for some $i, j \in [k]$, then replace the list of v by $\{i - 1, i + 1, i - 3, i + 3\} \cap L(v)$.

Clearly, any update creates an equivalent subinstance of (G, L). Note that in the graph homomorphism literature this operation is usually referred to as *edge (or arc) consistency* and it is performed in the beginning of most algorithms solving variants of H-Coloring [22, 30]. However, we keep the name "update" to emphasize that we will only perform it at certain points in our algorithm. We say that an update of v from w is *effective* if the size of the list of v decreases by at least 1, and *ineffective* otherwise. Note that an update is effective if and only if there exists an element $c \in L(v)$ which is not an element of $\{i - 1, i + 1\}$, $\{i, i + 2, i - 2\}$ or $\{i - 1, i + 1, i - 3, i + 3\}$ depending on the case in the definition of an update. We observe that the update does not change the goodness of the list[2].

▶ **Lemma 2.1** (♠). *If the lists of v and w are good before updating v from w, then the list of v is good or empty after the update.*

A C_k-list assignment L is said to be *reduced* if no effective update can be performed. It is well-known that one can obtain a reduced list assignment in polynomial time.

[2] The proofs of theorems and lemmas marked with ♠ are omitted due to space constraints.

▶ **Lemma 2.2** (♠)**.** *Let G be a graph of order n, and L be a C_k-list assignment. There exists an $O(n^3)$-time algorithm to obtain an equivalent reduced subinstance (G, L') of (G, L) or determine that (G, L) has no C_k-coloring.*

We now introduce two more tools that are important for our purpose. The first one is purely graph-theoretic and describes the structure of P_t-free graphs.

▶ **Theorem 2.3** ([6])**.** *Let G be a connected P_t-free graph with $t \geq 4$. Then G has a connected dominating set D such that $G|D$ is either P_{t-2}-free or isomorphic to P_{t-2}.*

The next observation generalizes the observation by Edwards [10] that List k-Coloring can be solved in polynomial time, whenever the size of each list is at most two. This was already noted by e.g. Feder and Hell [12].

▶ **Theorem 2.4** (♠)**.** *Let (G, L) be an instance of* List H-Coloring *where G is of order n and $|L(v)| \leq 2$ for every $v \in V(G)$. Then one can determine in $O(n^2)$ time if (G, L) is H-colorable and find an H-coloring if one exists.*

3 Polynomial algorithm for P_9-free graphs

In this section, we show that C_k-Precoloring Extension can be solved in polynomial time for P_9-free graphs.

▶ **Theorem 1.1.** *Let $k \geq 5$ be odd, G be a P_9-free graph of order n, W be a subset of its vertices, and h be a mapping from W to $V(C_k)$. Then one can determine in $O(n^{12k+3})$ time if h can be extended to a C_k-coloring of G, and find such a C_k-coloring if one exists.*

Outline of the proof

The overall strategy is to reduce the instance (G, W, h), in polynomial time, to a set $\mathcal{I}$ of polynomially many instances of List C_k-Coloring, in which every list has size at most 2, and (G, W, h) is an yes-instance if and only if at least one instance from $\mathcal{I}$ is a yes-instance. We then apply Theorem 2.4 to solve each instance from $\mathcal{I}$ in polynomial time.

More specifically, our algorithm, at a high level, consists of five phases. In the first three of them, we focus on processing the graph $G' := G \setminus W$. First, we apply Theorem 2.3 to show that the vertex set of G' can be partitioned into four sets (S, X, Y, Z) such S is connected and dominates X, X dominates Y, and Y dominates Z. Second, we branch on every possible C_k-coloring of $G'|S$. For each of these colorings of $G'|S$, we propagate the coloring of S to the vertices of $G' \setminus S$ via updates. After updating, the vertices in $S \cup X$ will have lists of size at most 2, but the vertices in $Y \cup Z$ may still have larger lists. In the third phase, we reduce the instance to polynomially many subinstances via branching in such a way that each of the subinstances avoids certain configurations, which we call *bad paths*. Finally, using the fact that each subinstance has no bad paths, in the last two phases we reduce the list size of vertices in $Y \cup Z$ to at most 2, restore the set W, creating a set of instances, which is equivalent to (G, W, h), and use Theorem 2.4 to solve the created instances in polynomial time.

Proof of Theorem 1.1. We view (G, W, h) as an equivalent instance (G, L) of List C_k-Coloring where $L(v) = \{h(v)\}$ if $v \in W$ and $L(v) = [k]$ otherwise.

Clearly, h can be extended to an C_k-coloring of G if and only if (G, L) is C_k-colorable. Moreover, observe that if G contains a triangle, then we can immediately report a no-instance. Checking for existence of triangles can clearly be done in $O(n^3)$ time, so from now on we assume that G is triangle-free.

For the first three phases we consider the graph $G' := G \setminus W$. We may assume that G' is connected, for otherwise we can apply the same reasoning to every connected component.

Phase I. Obtaining a layer structure

Let us start with imposing some structure on the vertices of G'.

▷ Claim 3.1. There exists $S \subseteq V(G')$ such that $|S| \leq 7$, the graph $G'|S$ is connected, and $S \cup N(S) \cup N(N(S))$ dominates G'.

Proof. We apply Theorem 2.3 to G': G' has a connected dominating set D that induces a subgraph that is either P_7-free or isomorphic to a P_7. If $G'|D$ is isomorphic to a P_7, then D is the desired set S. Otherwise we apply Theorem 2.3 on $G'|D$ to conclude that $G'|D$ has a connected dominating set D' that induces a subgraph that is either P_5-free or isomorphic to a P_5. If $G'|D'$ is isomorphic to a P_5, then D' is the desired set S. Otherwise $G'|D'$ is P_5-free. We again apply Theorem 2.3 on $G'|D'$: $G|D'$ has a connected dominating set D'' that induces a subgraph that is either P_3-free or isomorphic to P_3. Then $D'' \cup N(D'') \cup N(N(D''))$ dominates G'. Since G' is triangle-free, if $G'|D''$ is P_3-free, then D'' is a clique of size at most 2. It follows that $|D''| \leq 3$ and thus we can choose D'' for S. ◁

Let S be the set given by Claim 3.1. Define $X = N(S)$, $Y = N(N(S)) \setminus S$ and $Z = V(G') \setminus (X \cup Y \cup Z)$. Then (S, X, Y, Z) is a partition of $V(G')$, where S dominates X, X dominates Y and Y dominates Z, and there is no edge between S and $Y \cup Z$ or between X and Z. Moreover, S is connected. Such a quadruple $\mathcal{P} = (S, X, Y, Z)$ is called a *layer structure* of G'. The set S is called the *seed* of $\mathcal{P}$.

Phase II. Obtaining a canonical C_k-list assignment via updates

We now branch on every possible C_k-coloring of $G'|S$, there are at most k^7 such colorings since $|S| \leq 7$. Note that k^7 is a constant since k is a fixed number. To prove the theorem, therefore, it suffices to determine whether there is a branch, in which the precoloring of $S \cup W$ can be extended to a C_k-coloring of (G, L) in polynomial time.

In the following, we consider a fixed coloring $f : S \to [k]$, and we continue with the instance (G', L') of List C_k-Coloring, where $L'(v) = \{f(v)\}$ if $v \in S$ and $L'(v) = [k]$ otherwise.

We further partition the sets S, X, and Y as follows. For $1 \leq i \leq k$, we define

$$\begin{aligned} S_i :=& \{s \in S : L(s) = \{i\}\}, \\ X_i :=& \{x \in X \setminus (\bigcup_{j=1}^{i-1} X_j) : N(x) \cap S_i \neq \emptyset\}, \\ Y_i :=& \{y \in Y \setminus (\bigcup_{j=1}^{i-1} Y_j) : N(y) \cap X_i \neq \emptyset\}. \end{aligned}$$

Clearly, $(X_1, X_2, \ldots, X_k)$ is a partition of X and $(Y_1, Y_2, \ldots, Y_k)$ is a partition of Y.

We now perform the following updates for all $1 \leq i \leq k$ in the following order.

- For every edge sx with $s \in S_i$ and $x \in X_i$, we update x from s.
- For every edge xy with $x \in X_i$ and $y \in Y_i$, we update y from x.

We continue to denote the resulting C_k-list assignment by L'. Then $|L'(s)| = 1$ for every $s \in S$, $L'(x) \subseteq \{i-1, i+1\}$ for every $x \in X_i$ and $L'(y) \subseteq \{i, i-2, i+2\}$ for every $y \in Y_i$. We call such a C_k-list assignment L' *canonical* for $\mathcal{P} = (S, \bigcup_{i=1}^k X_i, \bigcup_{i=1}^k Y_i, Z)$.

▷ Claim 3.2 (♠). If X_i is not stable, then (G', L') is not C_k-colorable.

Note that one can determine in $O(n^2)$ time if there exists an X_i that is not stable. If so, we stop and correctly determine that (G', L') is not C_k-colorable by Claim 3.2. Otherwise, we may assume that X_i is stable for all $1 \le i \le k$ from now on.

Phase III. Eliminating bad paths via branching ($O(n^{12k})$ branches)

In this phase, we shall reduce the instance (G', L') to an equivalent set of polynomially many subinstances so that every subinstance has no bad paths, which we define now.

An induced path $a - b - c$ is a *bad path* in $\mathcal{P} = (S, X, Y, Z) = (S, \bigcup_{i=1}^k X_i, \bigcup_{i=1}^k Y_i, Z)$ if for some $i \in [k]$ we have $a \in Y_i$, $b, c \in (Y \cup Z) \setminus Y_i$, and $\{b, c\}$ is anticomplete to X_i. We call a the *starter* of $a - b - c$. Let $\mathcal{Q}_i$ be the set of all bad paths with starters in Y_i, clearly $|\mathcal{Q}_i| = O(n^3)$.

A vertex $v \in Y_i$ is of *depth at least* ℓ in $\mathcal{P}$ if for every $x \in N(v) \cap X_i$, there exists an induced path $v - x - P$ of length at least ℓ such that $V(P) \subseteq S$. Observe that every vertex in Y is of depth at least 3 to S (because we may assume that $|S| \ge 2$ and so no vertex in X is complete to S since G is triangle-free), and that the starter of a bad path is of depth at most 7 to S since G' is P_9-free.

Note that for any C_k-coloring of (G', L') and every $i \in [k]$, either there exists a bad path in $\mathcal{Q}_i$ whose starter is colored with a color in $\{i-2, i+2\}$ or the starters of all bad paths in $\mathcal{Q}_i$ are colored with i. This leads to the following branching scheme, which only updates the lists.

Branching.

- ($2^k = O(1)$ branches.) For every subset $I \subseteq [k]$, we have a branch B_I intended to find possible colorings such that there exists a bad path in $\mathcal{Q}_i$ whose starter is colored with a color in $\{i-2, i+2\}$ if $i \in I$, and all starters of bad paths in $\mathcal{Q}_i$ are colored with color i if $i \notin I$. Clearly, (G', L') is C_k-colorable if and only if at least one of the B_I is a yes-instance. In the following, we fix a branch B_I.
- ($O(2^k n^{3k}) = O(n^{3k})$ branches.) We further branch to obtain a set of size $O(n^{3k})$ of subinstances within B_I by guessing, for each $i \in I$, a bad path in $\mathcal{Q}_i$, and guessing the color of its starter from $\{i-2, i+2\}$. The union over all branches B_I of these subinstances is equivalent to (G', L').

  Specifically, for each element $(a_i - b_i - c_i)_{i \in I}$ in $\Pi_{i \in I} \mathcal{Q}_i$, we have one branch where we set $L''(a_i) := L'(a_i) \cap \{i-2, i+2\}$ for every $i \in I$, and we set $L''(a) := L'(a) \cap \{i\}$ for every starter a of a bad path in $\mathcal{Q}_i$ for every $i \notin I$. We denote the resulting C_k-list assignment by L''. For each such branch and for every element $(q_i)_{i \in I}$ in $\Pi_{i \in I} L''(a_i)$, we have one branch where $L''(a_i) = \{q_i\}$ for all $i \in I$. It follows that for all $i \in I$ and $x \in X_i \cap N(a_i)$, the only possible color for x is $(q_i + i)/2$, and so we set $L''(x) = \{(q_i + i)/2\}$. Since $L''(a_i) \subseteq \{i-2, i+2\}$ for all $i \in I$, it follows that there are $2^{|I|} \le 2^k$ branches. Let us fix one such branch and denote the resulting instance by (G', L'').
- ($O(k^{3k}) = O(1)$ branches.) We let I^* be the subset of $[k] \setminus I$ of indices i such that $\mathcal{Q}_i$ contains a bad path. For each $i \in I^*$, we choose a bad path $a_i - b_i - c_i$ in $\mathcal{Q}_i$ such that $|N(a_i) \cap X_i|$ is minimum, where the minimum is taken over all bad paths in $\mathcal{Q}_i$. Choose a vertex $x_i \in N(a_i) \cap X_i$ for each $i \in I^*$. Define

$$Q := \bigcup_{i \in I} \{b_i, c_i\} \cup \bigcup_{i \in I^*} \{b_i, c_i, x_i\},$$

where for $i \in I$, b_i, c_i are two vertices on the bad path we guessed in the previous bullet. We branch on every possible coloring of Q. Since $|Q| \leq 3k$, the number of branches is at most k^{3k}. In the following, we fix a coloring g of Q and denote the resulting subinstance by (G', L'''), where $L'''(v) = \{g(v)\}$ if $v \in Q$ and $L'''(v) = L''(v)$ otherwise.

Obtaining a new layer structure with a canonical C_k-list assignment. We now deal with (G', L'''). Define

$$A = \bigcup_{i \in I} ((N(a_i) \cap X_i) \cup \{a_i, b_i, c_i\}) \cup \bigcup_{i \in I^*} \{x_i, a_i, b_i, c_i\},$$

and note that in L''', every vertex in A has a list of size at most 1. We update all vertices of G' from all vertices in A and continue to denote the resulting C_k-list assignment by L'''. We now obtain a new partition $\mathcal{P}' = (S', X', Y', Z')$ of G' as follows.

- Let $S' := S \cup A$.
- For each $1 \leq j \leq k$, let $K_j := \emptyset$. For each vertex $v \in Y \cup Z$, if v has a neighbor in S', let j be the smallest integer in $[k]$ such that there exists a vertex $s \in N(v) \cap S'$ with $L(s) = \{j\}$, and add v to K_j. For each $1 \leq j \leq k$, let $X'_j = (X_j \cup K_j) \setminus A$. Let $X' := \bigcup_{i=1}^k X'_i$.
- For $1 \leq i \leq k$, let Y'_i be the set of vertices in $V(G') \setminus (S' \cup X' \cup (\bigcup_{j<i} Y'_j))$ that have a neighbor in X'_i. Let $Y' := \bigcup_{i=1}^k Y'_i$.
- Let $Z' := V(G') \setminus (S' \cup X' \cup Y')$.

▷ Claim 3.3 (♠). The new partition $\mathcal{P}' := (S', X', Y', Z')$ is a layer structure of G' and L''' is a canonical C_k-list assignment for $\mathcal{P}'$.

▷ Claim 3.4 (♠). For every $i \in [k]$ it holds that

1. $X'_i \setminus X_i \subseteq Y \cup Z$.
2. If a vertex in $Y' \cup Z'$ is anticomplete to X'_i, then it is anticomplete to X_i.
3. $Y'_i \setminus Y_i$ is anticomplete to X_i.

The following claim is the key to our branching algorithm.

▷ Claim 3.5. Let a be a starter of a bad path in $\mathcal{P}'$. If the depth of the starter of any bad path in $\mathcal{P}$ is at least ℓ, then the depth of a in $\mathcal{P}'$ is at least $\ell + 1$.

Proof. Let $a' - b' - c'$ be a bad path in $\mathcal{P}'$ with $a' \in Y'_i$. Consider the following cases.

Case 1: $a' \in Y_i \cap Y'_i$. Then $\emptyset \neq N(a') \cap X_i \subseteq X'_i$. By item 2. in Claim 3.4, $\{b', c'\}$ is anticomplete to X_i and so $a' - b' - c'$ is also a bad path in $\mathcal{P} = (S, X, Y, Z)$. This implies that $\mathcal{Q}_i \neq \emptyset$. Therefore, there exist $a, b, c, x \in S'$ such that $a - b - c$ is a bad path in $\mathcal{P}$ with $a \in Y_i$ and $x \in N(a) \cap X_i$.

We first claim that it is possible to pick a vertex $x' \in N(a') \cap X_i$ that is not adjacent to a. Recall that the branch we consider corresponds to a set $I \subseteq [k]$. If $i \in I$, then all vertices in $N(a) \cap X_i$ are in A and hence are now in S'. So every vertex in $N(a') \cap X_i$ is not adjacent to a, and our claim holds. If $i \notin I$, then $i \in I^*$, and so $a = a_i$. By the choice of a_i, it follows that $|N(a) \cap X_i| \leq |N(a') \cap X_i|$. Since $a' \in Y'_i$, it follows that a' is not adjacent to x. Therefore, there exists a vertex $x' \in N(a') \cap X_i$ such that x' is not adjacent to a.

Note that x and x' are not adjacent by Claim 3.2. Moreover, x' is anticomplete to $\{b', c', b, c\}$ by the definition of bad path. Let P' be the shortest path from x to x' with internal vertices contained in S. Note that P' exists since S is connected. Then P' is an induced path. Since $V(P') \setminus \{x, x'\} \subseteq S$, it follows that $V(P') \setminus \{x, x'\}$ is anticomplete to $\{a, b, c, a', b', c'\}$. Therefore, $c - b - a - x - P' - x' - a' - b' - c'$ is an induced path of order at least 9, a contradiction.

Case 2: $a' \in Y'_i \setminus Y_i$. It follows from Claim 3.4, item 3. that $N(a') \cap X'_i \subseteq X'_i \setminus X_i$. Pick a vertex $x' \in N(a') \cap X'_i$. Since $x \in X'_i \setminus X_i$, x' has a neighbor $s' \in S'$ by the definition of X'_i. By Claim 3.4, item 1., $x' \in Y \cup Z$ and so $s' \in S' \setminus S = A$. Thus there exists $j \in I$ such that x' is not anticomplete to $Q = \{x_j, a_j, b_j, c_j\}$, where $x_j \in N(a_j) \cap X_j$. Let $a_j - x_j - P$ be an induced path of length ℓ with $V(P) \subseteq S$. Note that $x' \in Y \cup Z$ is anticomplete to $V(P) \subseteq S$. Let $x' - P'' - x_j$ be the shortest path from x' to x_j such that $V(P'') \subseteq Q$. Since a' is anticomplete to $\{x\} \cup V(P) \cup V(P'') \subseteq S'$, it follows that $a' - x' - P'' - x_j - P$ is an induced path of length at least $\ell + 1$. This proves the claim. ◁

Therefore, we have obtained an equivalent set of subinstances of size $O(n^{3k})$. For each such subinstance, the minimum depth of the starter of a bad path has increased by at least 1 compared to $\mathcal{P}$ due to Claim 3.5. Note that the depth of any starter of a bad path in $\mathcal{P}$ is at least 3. Moreover, since G' is P_9-free, the depth of any starter of a bad path is at most 7.

By branching 4 times, therefore, we obtain an equivalent set of $O(n^{12k})$ subinstances such that each subinstance has no bad paths.

Phase IV. Reducing the list size of vertices in Z

Now we go back to processing the graph G. Let us fix an instance of List C_k-Coloring on G', created in the previous phase, and let (G, L) denote the instance obtained by restoring the vertices of W. By $\mathcal{P} = (S, X, Y, Z)$ we denote the layer structure of G' with no bad paths and L is canonical for $\mathcal{P}$. We first reduce the list size of vertices in Z.

▷ Claim 3.6 (♠). The set Z is stable and each $z \in Z$ has neighbors in at most one of $\{Y_1, Y_2, \dots, Y_k\}$.

▷ Claim 3.7 (♠). Let $z \in Z$ be anticomplete to W and have a neighbor in Y_i. If (G, L) has a C_k-coloring, then (G, L) has a C_k-coloring c such that $c(z) \in \{i-1, i+1\}$.

We now modify the lists of vertices in Z: let $L(z) := L(z) \cap \{i-1, i+1\}$ for every $z \in Z$ that is anticomplete to W and has a neighbor in Y_i. It follows from Claim 3.7 that the resulting list is equivalent to the original one. We still denote by the resulting list L.

Phase V. Reducing the list size of vertices in Y

We now apply Lemma 2.2 to obtain a reduced C_k-list assignment L'. Then (G, L') is an equivalent subinstance of (G, L). If $L'(v) = \emptyset$ for some $v \in V(G)$, we stop and report a no-instance. Define:

$$\begin{aligned}
S' &:= \{v \in V(G') : |L'(v)| = 1\},\\
X'_i &:= \{v \in V(G') \setminus S' : L'(v) \subseteq \{i-1, i+1\}\}, 1 \le i \le k,\\
Y'_i &:= \{v \in V(G') \setminus (S' \cup X' \cup \bigcup_{j<i} Y'_j) : L'(v) \subseteq \{i, i-2, i+2\}\}, 1 \le i \le k,\\
X' &:= \bigcup_{i=1}^{k} X'_i,\\
Y' &:= \bigcup_{i=1}^{k} Y'_i.
\end{aligned}$$

Note $W \subseteq S'$ and that $(S', X', Y', \emptyset)$ is almost a layer structure of G except that S' is not necessarily connected. It follows from the definition of (S', X', Y') and Lemma 2.1 that $L'(x) = \{i-1, i+1\}$ for every $x \in X'_i$ and $L'(y) = \{i-2, i, i+2\}$ for every $y \in Y'_i$. For $1 \leq i \leq k$, we partition Y'_i into two subsets Y^1_i and Y^2_i, where Y^2_i is the set of isolated vertices in $G|Y'_i$ and $Y^1_i = Y'_i \setminus Y^2_i$. We prove a few properties for S', X' and Y'.

▷ Claim 3.8 (♠). The following hold for S', X' and Y'.

1. $V(G) = S' \cup X' \cup Y'$.
2. For every $i \in [k]$ and $y \in Y'_i$, we have $N(y) \cap X' \subseteq X'_i$.
3. For every $i \in [k]$, we have $X_i \subseteq X'_i \cup S'$ and $Y'_i \subseteq Y_i$.
4. For every $i \in [k]$ and $y \in Y^2_i$, the vertex y is anticomplete to $Y' \setminus (Y^2_{i+1} \cup Y^2_{i-1})$.

▷ Claim 3.9. If (G, L') is C_k-colorable, then there exists a C_k-coloring c of (G, L') such that for each $1 \leq i \leq k$, $c(y) = i$ for all $y \in Y^2_i$ and $c(y) \in \{i-2, i+2\}$ for all $y \in Y^1_i$.

Proof. Suppose that c' is a C_k-coloring of (G, L'). Note that each $u \in Y^1_i$ has a neighbor $v \in Y^1_i$ by the definition. Since $L'(u), L'(v) \subseteq \{i-2, i, i+2\}$ and c' is a C_k-coloring of (G, L'), it follows that $c'(u) \neq i$ and $c'(v) \neq i$. So $c'(u) \in \{i-2, i+2\}$.

Let $u \in Y^2_i$. Note that u can only have neighbors in X'_i or in $Y^2_{i+1} \cup Y^2_{i-1}$ by item 2. and item 4. of Claim 3.8. Define $c : V(G) \to [k]$ such that $c(v) := i$ if $v \in Y^2_i$ and $c(v) := c'(v)$ if $v \notin \bigcup_{i=1}^k Y^2_i$.

Then c is a C_k-coloring of (G, L'), since $c'(x) \in \{i-1, i+1\}$ for every $x \in X'_i$. This completes the proof. ◁

Let us point out that the special treatment of the sets Y^1_i is needed only for the case $k = 5$. For $k > 5$, if one Y_i contains two adjacent vertices, one can observe that there is no way to color them. Thus we can immediately report a no-instance (or let it be reported when we solve the corresponding 2-Sat instance).

Let us now modify the lists as follows. For each $1 \leq i \leq k$ and each $y \in Y'_i$, let $L'(y) := L'(y) \cap \{i-2, i+2\}$ if $y \in Y^1_i$ and $L'(y) := L'(y) \cap \{i\}$ if $y \in Y^2_i$. By Claim 3.9, the new list assignment is equivalent to the original one. Now for each $v \in V(G)$ we have $|L'(v)| \leq 2$ and so Theorem 2.4 applies.

This completes the proof of correctness of our algorithm. Clearly, the most expensive part of our algorithm is Phase III where we branch into $O(n^{12k})$ subinstances. Since each subinstance can be constructed in $O(n^3)$ time by Lemma 2.2 and each 2-Sat instance can be solved in $O(n^2)$ time by Theorem 2.4, the total running time is $O(n^{12k+3})$. ◀

4 Hardness results

In this section we prove the following theorem.

▶ **Theorem 1.2.** *Let F be a connected graph. If F is not a subgraph of a subdivided claw, then for every odd $k \geq 5$ the C_k-Precoloring Extension problem is NP-complete for F-free graphs.*

We will prove Theorem 1.2 in several steps in which we analyze possible structure of F. We start with the following simple observation that will be repeatedly used. For the rest of this section, let $k = 2s+1$ for $s \geq 2$.

▶ **Observation 4.1.** *Let $s \geq 2$ be an integer and P be a $2s$-vertex path with endvertices a and b. Then the following holds.*

- *In any C_{2s+1}-coloring h of P we have $h(a) \neq h(b)$.*
- *For any distinct $i, j \in \{1, 2, \ldots, 2s+1\}$, there exists a C_{2s+1}-coloring h of P such that $h(a) = i$ and $h(b) = j$.*

4.1 Eliminate cycles

The *girth* of a graph G, denoted by girth(G), is the length of a shortest cycle in G. A vertex in a graph is called a *branch vertex* if its degree is at least 3. By Γ_p we denote the class of graphs, in which the number of edges in any path joining two branch vertices is divisible by p.

We first show that the problem is NP-hard in F-free graphs, unless F is a tree in Γ_{2s-1}.

▶ **Theorem 4.2.** *For each fixed integer $s \geq 2$ and each connected graph F, C_{2s+1}-*Coloring *is NP-complete for F-free graphs whenever F contains a cycle or is not in Γ_{2s-1}.*

Proof. It is known (see e.g. [31]) that the $(2s+1)$-Coloring problem is NP-complete for graphs of girth at least g for each fixed $g \geq 3$. We reduce this problem to C_{2s+1}-Coloring. Given a graph G, we obtain a graph G' by replacing each edge of G by a $(2s-1)$-edge path. Then it follows from Observation 4.1 that G is $(2s+1)$-colorable if and only if G' is C_{2s+1}-colorable. Clearly, $\text{girth}(G') = \text{girth}(G) \cdot (2s-1) \geq g(2s-1)$. Thus, if we choose $g \geq 3$ such that $g(2s-1) > \text{girth}(F)$, e.g., $g = |V(F)| + 1$, it follows that all graphs of girth at least $g(2s-1)$ are F-free. Moreover, it is easy to see that the number of edges in any path joining two branch vertices of G' is divisible by $2s-1$, so if $F \notin \Gamma_{2s-1}$, then G' does not contain F. ◀

4.2 Eliminate vertices of degree at least 4

From now on it suffices to consider trees with branch vertices at distance divisible by $2s-1$. We now show that C_k-Coloring is NP-complete for F-free graphs if F contains a vertex of degree at least 4. Note that in this case every subcubic graph is F-free.

▶ **Theorem 4.3 (♠).** *For each fixed $s \geq 2$, C_{2s+1}-*Coloring *is NP-complete for subcubic graphs.*

4.3 Eliminate multiple branch vertices

Before we prove the main theorem we need one more intermediate step that allows us to eliminate those F in which there are two branch vertices that are at distance not divisible by s. The proof is a reduction from the problem called Non-Rainbow Coloring Extension, whose instance is a 3-uniform hypergraph H and a partial coloring f of some of its vertices with colors $\{1, 2, 3\}$. We ask whether f can be extended to a 3-coloring of $V(H)$ such that no hyperedge is *rainbow* (i.e., contains three distinct colors). This problem is known to be NP-complete [3].

▶ **Theorem 4.4.** *For each fixed integer $s \geq 2$, C_{2s+1}-*Precoloring Extension *is NP-complete for bipartite graphs in Γ_s.*

Proof. We reduce from Non-Rainbow Coloring Extension. Let $H = (V, E)$ be a 3-uniform hypergraph and let f be a partial 3-coloring of H. We construct an instance of C_{2s+1}-Precoloring Extension as follows.

- For each vertex $v \in V$, we introduce a variable vertex, denoted by v'. If v is precolored by f, we precolor v' with the color $f(v)$.
- For each v that is not precolored by f, we introduce $2s-2$ new vertices and precolor them with $4, 5, \ldots, 2s+1$, respectively. Then each of these new vertices is joined by a $(2s-1)$-edge path to v'. It follows from Observation 4.1 that each vertex v' can only be mapped to one of $1, 2, 3$, and any of these three choices is possible.
- For each hyperedge $e = \{x, y, z\} \in E$, we add a new vertex v_e and three s-edge paths connecting v_e to x',y', and z', respectively. This whole subgraph is called an *edge gadget.*

Observe that if x' is mapped to $i \in \{1, 2, 3\}$, then the possible colors for v_e are $\{s+i, s+i-2, \ldots, s+i-2\lfloor s/2 \rfloor\} \cup \{s+i+1, s+i+3, \ldots, s+i+1+2\lfloor s/2 \rfloor\}$. Thus, if each of x', y', z' is mapped to a different vertex from $\{1, 2, 3\}$, then there is no way to extend this mapping to the whole edge gadget. On the other hand, such an extension is possible whenever x', y', z' receive at most two distinct colors.

We denote by G the resulting graph. By the properties of variable vertices and edge gadgets, (H, f) is an yes-instance of Non-Rainbow Coloring Extension if and only if the precoloring of G can be extended to a C_{2s+1}-coloring of G. Clearly, G is bipartite and belongs to Γ_s. ◀

By Theorems 4.2, 4.3, and 4.4, the C_{2s+1}-Precoloring Extension problem is NP-complete for F-free graphs unless F is a tree in $\Gamma_{s(2s-1)}$ (observe that s and $2s-1$ are relatively prime). We are now ready to show that the problem is NP-hard if F has more than one branch vertex.

▶ **Theorem 4.5.** *Let $s \geq 2$ be an integer and let F be a tree. If F contains two branch vertices, then C_{2s+1}-Coloring is NP-complete for F-free graphs.*

Proof. Let d be the distance between two closest branch vertices in F. We reduce from Positive Not-All-Equal Sat with all clauses containing exactly three literals. Consider an instance with variables $x_1, x_2, \ldots, x_n$ and clauses $D_1, D_2, \ldots, D_m$.

- We start our construction by introducing one special vertex z.
- For each variable x_i, we introduce a vertex v_i, adjacent to z.
- For each clause $D_\ell = \{x_i, x_j, x_k\}$, we introduce three new vertices $y_{\ell,i}$, $y_{\ell,j}$, and $y_{\ell,k}$, and join each pair of them with a $(2s-1)$-edge path. This guarantees that in every C_{2s+1}-coloring, they get three distinct colors. These three paths constitute the *clause gadget*.
- For each variable x_i belonging to a clause D_ℓ, we join each $y_{\ell,i}$ to v_i by a path $P_{\ell,i}$ with $2d(2s-1)+1$ edges. Let $v_i = p_1, p_2, \ldots, p_{2d(2s-1)+2} = y_{\ell,i}$ be the consecutive vertices of $P_{\ell,i}$. We add edges joining z and $p_{1+j(2s-1)}$ for every $1 \leq j \leq 2d$.

This completes the construction of a graph G. We claim that G is C_{2s+1}-colorable if and only if the initial formula is satisfiable, and that G belongs to our class.

▷ Claim 4.6 (♠). G is C_{2s+1}-colorable if and only if the initial formula is satisfiable.

▷ Claim 4.7 (♠). G is F-free.

This completes the proof of Theorem 4.5. ◀

Now Theorem 1.2 comes from combining the Theorems 4.2, 4.3, 4.4, and 4.5. We observe that all reductions in our hardness proofs are linear in the number of vertices (the target graph is assumed to be fixed, so s is a constant). Moreover, all problems we are reducing from can be shown to be NP-complete by a linear reduction from 3-Sat. Thus we get the following result, conditioned on the Exponential Time Hypothesis (ETH), which, along with the *sparsification lemma*, implies that 3-Sat with n variables and n clauses cannot be solved in time $2^{o(n+m)}$ [27, 28].

▶ **Corollary 4.8.** *Unless the ETH fails, the following holds. If F is a connected graph that is not a subgraph of a subdivided claw, then for every $s \geq 2$, the C_{2s+1}-Precoloring Extension problem cannot be solved in time $2^{o(n)}$ in F-free graphs with n vertices.*

5 Conclusion

In this paper, we initiate a study of C_{2s+1}-Coloring for F-free graphs for a fixed graph F. We prove that C_{2s+1}-Precoloring Extension is NP-complete for F-free graphs if some component of F is not a subdivided claw. Moreover, we show that C_{2s+1}-Precoloring Extension is polynomial-time solvable for P_9-free graphs. Note that all our hardness results work for C_{2s+1}-Coloring, except for Theorem 4.4. Thus it is natural to ask whether analogous hardness results holds for C_{2s+1}-Coloring too. Moreover, the following questions seem natural to explore.

- Are there values of s and t such that C_{2s+1}-Coloring is NP-complete for P_t-free graphs?
- Is C_{2s+1}-Coloring polynomial for F-free graphs when F is a subdivided claw?
- Is C_{2s+1}-Coloring FPT for P_t-free graphs, when parameterized by s?

References

1 Vladimir E Alekseev. The effect of local constraints on the complexity of determination of the graph independence number. *Combinatorial-algebraic methods in applied mathematics*, pages 3–13, 1982.

2 Gábor Bacsó, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Zsolt Tuza, and Erik Jan van Leeuwen. Subexponential-Time Algorithms for Maximum Independent Set in P_t-Free and Broom-Free Graphs. *Algorithmica*, 81(2):421–438, 2019. doi:10.1007/s00453-018-0479-5.

3 Manuel Bodirsky, Jan Kára, and Barnaby Martin. The complexity of surjective homomorphism problems – a survey. *Discrete Applied Mathematics*, 160(12):1680–1690, 2012. doi:10.1016/j.dam.2012.03.029.

4 Flavia Bonomo, Maria Chudnovsky, Peter Maceli, Oliver Schaudt, Maya Stein, and Mingxian Zhong. Three-Coloring and List Three-Coloring of Graphs Without Induced Paths on Seven Vertices. *Combinatorica*, 38(4):779–801, 2018. doi:10.1007/s00493-017-3553-8.

5 Andrei A. Bulatov. A Dichotomy Theorem for Nonuniform CSPs. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 319–330. IEEE Computer Society, 2017. doi:10.1109/FOCS.2017.37.

6 Eglantine Camby and Oliver Schaudt. A New Characterization of P_k-Free Graphs. *Algorithmica*, 75(1):205–217, 2016. doi:10.1007/s00453-015-9989-6.

7 Maria Chudnovsky, Sophie Spirkl, and Mingxian Zhong. Four-coloring P_6-free graphs. I. Extending an excellent precoloring. *CoRR*, abs/1802.02282, 2018. arXiv:1802.02282.

8 Maria Chudnovsky, Sophie Spirkl, and Mingxian Zhong. Four-coloring P_6-free graphs. II. Finding an excellent precoloring. *CoRR*, abs/1802.02283, 2018. arXiv:1802.02283.

9 Maria Chudnovsky, Sophie Spirkl, and Mingxian Zhong. Four-coloring P_6-free graphs. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 1239–1256. SIAM, 2019. doi:10.1137/1.9781611975482.76.

10 Keith Edwards. The Complexity of Colouring Problems on Dense Graphs. *Theor. Comput. Sci.*, 43:337–343, 1986. doi:10.1016/0304-3975(86)90184-2.

11 Jessica Enright, Lorna Stewart, and Gábor Tardos. On List Coloring and List Homomorphism of Permutation and Interval Graphs. *SIAM J. Discrete Math.*, 28(4):1675–1685, 2014. doi:10.1137/13090465X.

12 Tomas Feder and Pavol Hell. List Homomorphisms to Reflexive Graphs. *J. Comb. Theory, Ser. B*, 72(2):236–250, 1998. doi:10.1006/jctb.1997.1812.

13 Tomás Feder, Pavol Hell, and Jing Huang. List Homomorphisms and Circular Arc Graphs. *Combinatorica*, 19(4):487–505, 1999. doi:10.1007/s004939970003.

14 Tomás Feder, Pavol Hell, Sulamita Klein, Loana Tito Nogueira, and Fábio Protti. List matrix partitions of chordal graphs. *Theor. Comput. Sci.*, 349(1):52–66, 2005. doi:10.1016/j.tcs.2005.09.030.

15 Tomás Feder and Moshe Y. Vardi. Monotone monadic SNP and constraint satisfaction. In S. Rao Kosaraju, David S. Johnson, and Alok Aggarwal, editors, *Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing, May 16-18, 1993, San Diego, CA, USA*, pages 612–622. ACM, 1993. doi:10.1145/167088.167245.

16 Anna Galluccio, Pavol Hell, and Jaroslav Nesetril. The complexity of H-colouring of bounded degree graphs. *Discrete Mathematics*, 222(1-3):101–109, 2000. doi:10.1016/S0012-365X(00)00009-1.

17 Petr A. Golovach, Matthew Johnson, Daniël Paulusma, and Jian Song. A Survey on the Computational Complexity of Coloring Graphs with Forbidden Subgraphs. *Journal of Graph Theory*, 84(4):331–363, 2017. doi:10.1002/jgt.22028.

18 Petr A. Golovach, Daniël Paulusma, and Jian Song. Closing complexity gaps for coloring problems on H-free graphs. *Inf. Comput.*, 237:204–214, 2014. doi:10.1016/j.ic.2014.02.004.

19 Carla Groenland, Karolina Okrasa, Paweł Rzążewski, Alex Scott, Paul Seymour, and Sophie Spirkl. H-colouring P_t-free graphs in subexponential time. *Discrete Applied Mathematics*, (to appear), 2019.

20 Andrzej Grzesik, Tereza Klimošová, Marcin Pilipczuk, and Michal Pilipczuk. Polynomial-time algorithm for Maximum Weight Independent Set on P_6-free graphs. *CoRR*, abs/1707.05491, 2017. arXiv:1707.05491.

21 Andrzej Grzesik, Tereza Klimošova, Marcin Pilipczuk, and Michal Pilipczuk. Polynomial-time algorithm for Maximum Weight Independent Set on P_6-free graphs. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 1257–1271. SIAM, 2019. doi:10.1137/1.9781611975482.77.

22 Pavol Hell and Jaroslav Nesetril. *Graphs and Homomorphisms*. Oxford University Press, July 2004. doi:10.1093/acprof:oso/9780198528173.001.0001.

23 Pavol Hell and Jaroslav Nešetřil. On the complexity of H-coloring. *J. Comb. Theory, Ser. B*, 48(1):92–110, 1990. doi:10.1016/0095-8956(90)90132-J.

24 Chính T. Hoàng, Marcin Kamiński, Vadim V. Lozin, Joe Sawada, and Xiao Shu. Deciding k-Colorability of P_5-Free Graphs in Polynomial Time. *Algorithmica*, 57(1):74–81, 2010. doi:10.1007/s00453-008-9197-8.

25 Ian Holyer. The NP-Completeness of Edge-Coloring. *SIAM J. Comput.*, 10(4):718–720, 1981. doi:10.1137/0210055.

26 Shenwei Huang. Improved complexity results on k-coloring P_t-free graphs. *Eur. J. Comb.*, 51:336–346, 2016. doi:10.1016/j.ejc.2015.06.005.

27 Russell Impagliazzo and Ramamohan Paturi. On the Complexity of k-SAT. *Journal of Computer and System Sciences*, 62(2):367–375, 2001. doi:10.1006/jcss.2000.1727.

28 Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which Problems Have Strongly Exponential Complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001. doi:10.1006/jcss.2001.1774.

29 Jan Kratochvíl. Precoloring extension with fixed color bound. *Acta Mathematica Universitatis Comenianae. New Series*, 62, January 1993.

30 Benoit Larose and Adrien Lemaître. List-homomorphism problems on graphs and arc consistency. *Discrete Mathematics*, 313(22):2525–2537, 2013. doi:10.1016/j.disc.2013.07.018.

31 Vadim V. Lozin and Marcin Kamiński. Coloring edges and vertices of graphs without short or long cycles. *Contributions to Discrete Mathematics*, 2(1), 2007. URL: http://cdm.ucalgary.ca/cdm/index.php/cdm/article/view/60.

32 Gerhard J. Woeginger and Jirí Sgall. The complexity of coloring graphs without long induced paths. *Acta Cybern.*, 15(1):107–117, 2001. URL: http://www.inf.u-szeged.hu/actacybernetica/edb/vol15n1/Woeginger_2001_ActaCybernetica.xml.

33 Dmitriy Zhuk. A Proof of CSP Dichotomy Conjecture. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 331–342. IEEE Computer Society, 2017. doi:10.1109/FOCS.2017.38.