Choosability of P_5-free graphs*

Petr A. Golovach† Pinar Heggernes†

Abstract

A graph is k-choosable if it admits a proper coloring of its vertices for every assignment of k (possibly different) allowed colors to choose from for each vertex. It is NP-hard to decide whether a given graph is k-choosable for $k \geq 3$, and this problem is considered strictly harder than the k-coloring problem. Only few positive results are known on input graphs with a given structure. Here, we prove that the problem is fixed parameter tractable on P_5-free graphs when parameterized by k. This graph class contains the well known and widely studied class of cographs. Our result is surprising since the parameterized complexity of k-coloring is still open on P_5-free graphs. To give a complete picture, we show that the problem remains NP-hard on P_5-free graphs when k is a part of the input.

1 Introduction

Graph coloring is one of the most well known and intensively studied problems in graph theory. The k-Coloring problem asks whether the vertices of an input graph G can be colored with k colors such that no pair of adjacent vertices receive the same color (such coloring is also called a proper coloring). This problem is known to be NP-complete even when $k \geq 3$ is not a part of the input but a fixed constant.

Vizing [19] and Erdős et al. [6] introduced a version of graph coloring called list coloring. In list coloring, a set $L(v)$ of allowed colors is given for each vertex v of the input graph, and we want to decide whether a proper coloring of the graph exists such that each vertex v receives a color from $L(v)$. If G has a list coloring for every assignment of lists of cardinality k to its vertices, then G is said to be k-choosable. Hence the k-Choosability problem

*This work is supported by the Research Council of Norway.

†Department of Informatics, University of Bergen, N-5020 Bergen, Norway. Emails: {Peter.Golovach|Pinar.Heggernes}@ii.uib.no

asks whether an input graph G is k-choosable. List coloring has received increasing attention since the beginning of 90's, and there are very good surveys [1, 17] and books [11] on the subject. It is proved to be a very difficult problem; Gutner and Tarsi [9] proved that k-CHOOSABILITY is Π_2^P-complete for bipartite graphs for any fixed $k \geq 3$, whereas 2-CHOOSABILITY can be solved in polynomial time [6]. The 3-CHOOSABILITY and 4-CHOOSABILITY problems remain Π_2^P-complete for planar graphs, whereas any planar graph is 5-choosable [16]. Due to these hardness results, upto the assumption that NP is not equal to co-NP, CHOOSABILITY is strictly harder than COLORING on general graphs [1].

Despite being a difficult problem to deal with, CHOOSABILITY has applications in a large variety of areas, like various kinds of scheduling problems, VLSI design, and frequency assignments [1]. Consequently, any attempt to solve this problem is of interest, and we attack it using structural information on the input and parameterized algorithms. A problem is fixed parameter tractable (FPT) if its input can be partitioned into a main part (typically the input graph) of size n and a parameter (typically an integer) k so that there is an algorithm that solves the problem in time $O(n^c \cdot f(k))$, where f is a computable function dependent only on k, and c is a fixed constant independent of input [5]. In this case, we say that the problem is FPT when parameterized by k. The field of parameterized algorithms and fixed parameter complexity/tractability has been flourishing during the last decade, with many new results appearing every year in high level conferences and journals, and it has been enriched by several new books [7, 14].

In this paper, we show that k-CHOOSABILITY is fixed parameter tractable on P_5-free graphs. These are graphs containing no induced copy of a simple path on 5 vertices, and this graph class contains the class of cographs that has been subject to extensive theoretical study [3]. An interesting point to mention is that the fixed parameter tractability of k-COLORING on P_5-free graphs is still open [10]. As mentioned above, CHOOSABILITY is more difficult than COLORING on general graphs. Our result indicates that the opposite might be true for the class of P_5-free graphs. In last year's MFCS, Hoàng et al. showed that k-COLORING can be solved in polynomial time for any fixed k on P_5-free graphs [10], but in their running time k contributes to the degree of the polynomial. Furthermore, k-COLORING is NP-complete on P_5-free graphs when k is a part of input [12]. To give a complete picture, here we show that k-CHOOSABILITY is NP-hard on P_5-free graphs when k is a part of input. Thus fixed parameter tractability is the best we can expect to achieve for k-CHOOSABILITY on this graph class.

To mention other existing results on the coloring problem on graphs that do not contain long induced paths, 3-Coloring has a polynomial-time solution on P_6-free graphs [15], 5-Coloring is NP-complete for P_8-free graphs, and 4-Coloring is NP-complete for P_{12}-free graphs [20].

2 Definitions and preliminaries

We consider finite undirected graphs without loops or multiple edges. A graph is denoted by $G = (V, E)$, where $V = V(G)$ is the set of vertices and $E = E(G)$ is the set of edges. For a vertex $v \in V$, the set of vertices that are adjacent to v is called the *neighborhood* of v and denoted by $N_G(v)$ (we may omit index if the graph under consideration is clear from the context). The *degree* of a vertex v is $deg(v) = |N(v)|$. The *average degree* of G is $d(G) = \frac{1}{|V|}\sum_{v\in V} deg(v)$. For a vertex subset $U \subseteq V$ the subgraph of G induced by U is denoted by $G[U]$. A set $U \subseteq V$ is a *clique* if all vertices in U are pairwise adjacent in G. A set of vertices U is a *dominating set* if for each vertex $v \in V$, either $v \in U$ or there is a vertex $u \in U$ such that $v \in N(u)$. We also say that a subgraph H of G is dominating if $V(H)$ is a dominating set. We denote by $G - U$ the graph $G[V \setminus U]$, and by $G - u$ the graph $G[V \setminus \{u\}]$ for $u \in V$.

A *vertex coloring* of a graph $G = (V, E)$ is an assignment $c\colon V \to \mathbb{N}$ of a positive integer (*color*) to each vertex of G. The coloring c is *proper* if adjacent vertices receive distinct colors. Assume that each vertex $v \in V$ is assigned a *color list* $L(v) \subset \mathbb{N}$, which is the set of admissible colors for v. A mapping $c\colon V \to \mathbb{N}$ is a *list coloring* of G if c is a proper vertex coloring and $c(v) \in L(v)$ for every $v \in V$. For a positive integer k, G is *k-choosable* if G has a list coloring for every assignment of color lists $L(v)$ with $|L(v)| = k$ for all $v \in V$. The *choice number* (also called *list chromatic number*) of G, denoted $ch(G)$, is the minimum integer k such that G is k-choosable. The k-Choosability problem asks for a given graph G and a positive integer k, whether G is k-choosable. It is known that dense graphs have large choice number [1], as indicated by the following result.

Proposition 1 ([1]). *Let G be a graph and s be an integer. If*

$$d(G) > 4\binom{s^4}{s}\log(2\binom{s^4}{s})$$

then $ch(G) > s$.

By P_n we denote the graph on vertex set $\{v_1, v_2, \ldots, v_n\}$ and edge set $\{v_1v_2, v_2v_3, \ldots, v_{n-1}v_n\}$. A graph is *$P_n$-free* if it does not contain P_n as an induced subgraph. *Cographs* are the class of P_4-free graphs, and they are contained in the class of P_5-free graphs. These graph classes can be recognized in polynomial time. The following structural property of P_5-free graphs was proved by Bacsó and Tuza [2].

Proposition 2 ([2]). *Every connected P_5-free graph has either a dominating clique or a dominating P_3.*

It follows from the results of [2] that such a clique or path can be constructed in polynomial time.

Finally, we distinguish between the parameterized and the non-parameterized versions of our problem. In the Choosability problem, G and k are input. We denote by k-Choosability the version of the problem parameterized by k.

3 k-Choosability is FPT on P_5-free graphs

In this section we prove that k-Choosability is fixed parameter tractable on P_5-free graphs.

Theorem 1. *The k-Choosability problem is* FPT *on P_5-free graphs.*

Proof. We give a constructive proof of this theorem by describing a recursive algorithm based on Propositions 1 and 2 that checks whether $ch(G) \leq k$. We assume that $k \geq 3$, since for $k \leq 2$, k-Choosability can be solved in polynomial time for general graphs [6]. If G is disconnected, then $ch(G)$ is equal to the maximum choice number of the connected components of G. Thus we also assume that G is connected.

Our algorithm uses as its main tool a procedure called `Color`, given in Algorithm 1. This procedure takes as input a connected P_5-free graph G and a set $W = \{w_1, \dots, w_r\} \subseteq V(G)$ with a sequence of color lists $\mathcal{L} = (L(w_1), \dots, L(w_r))$, each of size k. For the notation in this procedure, we let $L = L(w_1) \cup \dots \cup L(w_r)$, and we denote $l = \max\{\max L(w_1), \dots, \max L(w_r)\}$. Let also $\mathbb{L} = L(w_1) \times \dots \times L(w_r)$ and $\mathbb{X} = 2^{\mathbb{L}}$. We say that vertices $w_1, \dots, w_r$ are colored by $c = (c_1, \dots, c_r) \in \mathbb{L}$ if each w_i is colored by c_i. Set $H = G - W$. Procedure `Color` produces an output which either contains a list of different sets $\mathcal{X} = (X_1, \dots, X_s)$, $X_i \in \mathbb{X}$, such that for any assignment of color lists of size k to vertices of H, there is a set X_i with the property that any $c \in X_i$ can be used for coloring of W with respect to adjacencies between vertices in W and vertices in $V(H)$, or the output contains "NO" if there is a list assignment for vertices of H such that no list coloring exists. Denote

Procedure `Color`(G, W, $\mathcal{L}$)

Find a dominating set $U = \{u_1, \ldots, u_p\}$ **of** $H = G - W$**, such that** U **is a clique or** U **induces a** P_3**;**

Let $\mathcal{X} = \emptyset$;

if $p > k$ **then** `Return`(**NO**), `Halt`;

if $d(G[W \cup U]) > d$ **then** `Return`(**NO**), `Halt`;

forall *Color lists* $L(u_1), \ldots, L(u_p) \subseteq \{1, \ldots, l, l+1, \ldots, l+kp\}$, *s.t.* $|L(u_i)| = k$ **do**

- **if** $U = V(H)$ **then**
  - **Let** $X = \emptyset$;
  - **forall** *List colorings* s *of* H **do**
    - **Let** $X := X \cup \{c \in \mathbb{L} \colon c(w_i) \neq s(u_j)$ **if** $w_i u_j \in E(G)\}$;
  - **if** $X \neq \emptyset$ **then** `Add`($\mathcal{X}, X$); **else** `Return`(**NO**), `Halt`;
- **if** $U \neq V(H)$ **then**
  - **Let** $H_1, \ldots, H_q$ **be the connected components of** $H - U$**, and let** $F_i = G[W \cup U \cup V(H_i)]$ **for** $i \in \{1, \ldots, q\}$;
  - **Let** $\mathcal{L}' = (L(u_1), \ldots, L(u_p))$, $\mathbb{L}' = \mathbb{L} \times L(u_1) \times \cdots \times L(u_p)$;
  - **for** $i = 1$ **to** q **do**
    - `Color`($F_i, W \cup U, \mathcal{L} \cup \mathcal{L}'$);
    - **if** *the output is* `NO` **then**
      - `Return`(**NO**), `Halt`;
    - **else**
      - **Let** $\mathcal{X}_i$ **be the output;**
  - **Let** $\mathcal{Y} = \mathcal{X}_1$;
  - **for** $i = 2$ **to** q **do**
    - **Let** $\mathcal{Z} = \emptyset$;
    - **forall** $X \in \mathcal{X}_i$ *and* $Y \in \mathcal{Y}$ **do**
      - **if** $X \cap Y \neq \emptyset$ **then** `Add`($\mathcal{Z}, X \cap X'$);
      - **else** `Return`(**NO**), `Halt`;
    - **Let** $\mathcal{Y} = \mathcal{Z}$;
  - **forall** $Z \in \mathcal{Z}$ **do**
    - **Let** $X = \{(c(w_1), \ldots, c(w_r)) \colon c \in Z, c(w_i) \neq c(u_j)$ **if** $w_i u_j \in E(G)$ **and** $c(u_i) \neq c(u_j)$ **if** $u_i u_j \in E(G)\}$;
    - **if** $X \neq \emptyset$ **then** `Add`($\mathcal{X}, X$);
    - **else** `Return`(**NO**), `Halt`;

if $\mathcal{X} = \emptyset$ **then** `Return`(**NO**), `Halt`; **else** `Return`($\mathcal{X}$).

Algorithm 1: Pseudo code for the procedure `Color`

$d = 4\binom{k^4}{k}\log(2\binom{k^4}{k})$. The subroutine $\texttt{Add}(A, a)$ adds the element a to the set A if $a \notin A$, and the subroutine $\texttt{Halt}$ stops the algorithm. Our main algorithm calls Procedure $\texttt{Color}(G, \emptyset, \emptyset)$. To simplify the description of the algorithm it is assumed that for $W = \emptyset$, $\mathbb{L}$ contains unique *zero* coloring (i.e. $\mathbb{L}$ is non empty). If the output is "NO" then G is not k-choosable, and otherwise G is k-choosable.

To prove the correctness of the algorithm, let us analyze one call of Procedure $\texttt{Color}$. Since each induced subgraph of a P_5-free graph is P_5-free, by Proposition 2 it is possible to construct the desired dominating set U in the beginning of the procedure. If $|U| > k \geq 3$ then U is a clique in G and $ch(G) \geq ch(G[U]) > k$. If $d(G[W \cup U]) > d$ then $ch(G) \geq ch(G[W \cup U]) > k$ by Proposition 1. Otherwise we proceed and consider color lists for vertices of U. It should be observed here that it is sufficient to consider only color lists with elements from the set $L \cup \{l+1, \ldots, l+kp\}$, since we have to take into account only intersections of these lists which each other and with lists for vertices of W. If $U = V(H)$ then the output is created by checking all possible list colorings of H. If $U \neq V(H)$ then we proceed with our decomposition of G. Graphs $F_1, \ldots, F_q$ are constructed and Procedure $\texttt{Color}$ is called recursively for them. It is possible to consider these graphs independently since vertices of different graphs H_i and H_j are not adjacent. Then outputs for $F_1, \ldots, F_q$ are combined and the output for G is created by checking all possible list colorings of U.

Now we analyze the running time of this algorithm. To estimate the depth of the recursion tree we assume that h sets U are created recursively without halting and denote them by $U_1, \ldots, U_h$. Since $|U_i| \leq k$, $|U_1 \cup \cdots \cup U_h| \leq kh$. Notice that each set U_i is a dominating set for $U_{i+1}, \ldots, U_h$. Hence $\sum_{v \in U_i} \deg_F(v) \geq h-1$, where $F = G[U_1 \cup \cdots \cup U_h]$, and $\sum_{v \in V(F)} \deg(v) \geq h(h-1)$. This means that $d(F) \geq \frac{h-1}{k}$, and if $h > kd+1 = 4k\binom{k^4}{k}\log(2\binom{k^4}{k}) + 1$ then Procedure $\texttt{Color}$ stops. Therefore the depth of the recursion tree is at most $kd+1 = 4k\binom{k^4}{k}\log(2\binom{k^4}{k}) + 1$. It can be easily noted that the number of leaves in the recursion tree is at most $n = |V(G)|$, and the number of calls of $\texttt{Color}$ is at most $(4k\binom{k^4}{k}\log(2\binom{k^4}{k})+1)n = O(k^5 \cdot 2^{k^4} \cdot n)$. Let us analyze the number of operations used for each call of this procedure. The set U can be constructed in polynomial time by the results of [2]. If $|U| > k$ then the algorithm finishes its work. Assume that $|U| \leq k$. Since the depth of the recursion tree is at most $kd+1$, color lists for vertices of U are chosen from the set $\{1, \ldots, (kd+1)k^2\}$, and the number of all such sets is $\binom{(kd+1)k^2}{k}$. So, there are at most $\binom{(kd+1)k^2}{k}^k$ (or $2^{O(k^8 \cdot 2^{k^4})}$) possibilities to assign color lists to vertices of U. The number of all list colorings of vertices of U is at most k^k. Recall that the output of `Color` is either "NO" or a list of different sets $\mathcal{X} = (X_1, \ldots, X_s)$ where $X_i \in \mathbb{X}$. Since the depth of the recursion tree is at most $kd + 1$ and each set U contains at most k elements (if the algorithm does not stop), the size of W is at most $k(kd+1)$. Hence the output contains at most $2^{k(kd+1)}$ (or $2^{O(k^6 \cdot 2^{k^4})}$) sets. Using these bounds and the observation that $q \leq n$, we can conclude that the number of operations for each call of `Color` is $2^{O(k^8 \cdot 2^{k^4})} \cdot n^c$ for some positive constant c. Taking into account the total number of calls of the procedure we can bound the the running time of our algorithm as $2^{O(k^8 \cdot 2^{k^4})} \cdot n^s$ for some positive constant s. □

4 Choosability is NP-hard on P_5-free graphs

In this section we show that Choosability, with input G and k, remains NP-hard when the input graph is restricted to P_5-free graphs.

Theorem 2. *The* Choosability *problem is* NP*-hard on* P_5*-free graphs.*

Proof. We reduce the not-all-equal 3-Satisfiability (NAE 3-SAT) problem with only positive literals [8] to Choosability. For a given set of Boolean variables $X = \{x_1, \ldots, x_n\}$, and a set $C = \{C_1, \ldots, C_m\}$ of three-literal clauses over X in which *all literals are positive*, this problem asks whether there is a truth assignment for X such that each clause contains at least one true literal and at least one false literal. NAE 3-SAT is NP-complete [8].

Our reduction has two stages. First we reduce NAE 3-SAT to List Coloring by constructing a graph with color lists for its vertices. Then we build on this graph to complete the reduction from NAE 3-SAT to Choosability.

At the first stage of the reduction we construct a complete bipartite graph ($K_{n,2m}$) H with the vertex set $\{x_1, \ldots, x_n\} \cup \{C_1^{(1)}, \ldots, C_m^{(1)}\} \cup \{C_1^{(2)}, \ldots, C_m^{(2)}\}$, where $\{x_1, \ldots, x_n\}$ and $(\{C_1^{(1)}, \ldots, C_m^{(1)}\} \cup \{C_1^{(2)}, \ldots, C_m^{(2)}\})$ is the bipartition of the vertex set. Hence on the one side of bipartition we have a vertex for each variable, and on the other side we have two vertices for each clause. We define color lists for vertices of H as follows: $L(x_i) = \{2i-1, 2i\}$ for $i \in \{1, \ldots, n\}$, $L(C_j^{(1)}) = \{2p-1, 2q-1, 2r-1\}$ and $L(C_j^{(2)}) = \{2p, 2q, 2r\}$ if the clause C_j contains literals x_p, x_q, x_r for $j \in \{1, \ldots, m\}$.

Lemma 1. *The graph H has a list coloring if and only if there is a truth assignment for the variables in X such that each clause contains at least one true literal and at least one false literal.*

Proof. Assume that H has a list coloring. Set the value of variable $x_i = true$ if vertex x_i is colored by $2i-1$, and set $x_i = false$ otherwise. For each clause C_j with literals x_p, x_q, x_r, at least one literal has value *true* since at least one color from the list $\{2p, 2q, 2r\}$ is used for coloring vertex $C_j^{(2)}$, and at least one literal has value *false*, since at least one color from the list $\{2p-1, 2q-1, 2r-1\}$ is used for coloring vertex $C_j^{(1)}$.

Suppose now that there is a truth assignment for the variables in X such that each clause contains at least one true literal and at least one false literal. For each variable x_i, we color vertex x_i by the color $2i-1$ if $x_i = true$, and we color x_i by the color $2i$ otherwise. Then any two vertices $C_j^{(1)}$ and $C_j^{(2)}$, which correspond to the clause C_j with literals x_p, x_q, x_r, can be properly colored, since at least one color from each of lists $\{2p-1, 2q-1, 2r-1\}$ and $\{2p, 2q, 2r\}$ is not used for coloring of vertices $x_1, \ldots, x_n$. □

Now we proceed with our reduction and add to H a clique with $k = n + 4nm - 4m$ vertices $u_1, \ldots, u_k$. For each vertex x_i, we add edges $x_i u_\ell$ for $\ell \in \{1, \ldots, k\}$, $\ell \neq 2i-1, 2i$. For vertices $C_j^{(1)}$ and $C_j^{(2)}$ which correspond to clause C_j with literals x_p, x_q, x_r, edges $C_j^{(1)} u_\ell$ such that $\ell \neq 2p-1, 2q-1, 2r-1$ and edges $C_j^{(2)} u_\ell$ such that $\ell \neq 2p, 2q, 2r$ are added for $\ell \in \{1, \ldots, k\}$. We denote the obtained graph by G.

We claim that G is k-choosable if and only if there is a truth assignment for the variables in X such that each clause contains at least one true literal and at least one false literal.

For the first direction of the proof of this claim, suppose that for any truth assignment there is a clause all of whose literals have the same value. Then we consider a list coloring for G with same color list $\{1, \ldots, k\}$ for each vertex. Assume without loss of a generality that u_i is colored by color i for $i \in \{1, \ldots, k\}$. Then each vertex x_i can be colored only by colors $2i-1, 2i$, each vertex $C_j^{(1)}$ can be colored only by colors $2p-1, 2q-1, 2r-1$ and each vertex $C_j^{(2)}$ can be colored only by colors $2p, 2q, 2r$ if $C_j^{(1)}, C_j^{(2)}$ correspond to the clause with literals x_p, x_q, x_r. By Lemma 1, it is impossible to extend the coloring of vertices $u_1, \ldots, u_k$ to a list coloring of G.

For the other direction, assume now that there is a truth assignment for the variables in X such that each clause contains at least one true literal and at least one false literal. Assign arbitrarily a color list $L(v)$ of size k to each vertex $v \in V(G)$. We show how to construct a list coloring of G. Denote by U the set of vertices $\{u_{2n+1}, \ldots, u_k\}$. Notice that U is a clique whose vertices are adjacent to all vertices of G. We start coloring the vertices of U and reducing G according to this coloring, using following rules:

1. If there is a non colored vertex $v \in U$ such that $L(v)$ contains a color c which was not used for coloring the vertices of U and there is a vertex $w \in \{x_1, \ldots, x_n\} \cup \{C_1^{(1)}, \ldots, C_m^{(1)}\} \cup \{C_1^{(2)}, \ldots, C_m^{(2)}\}$ such that $c \notin L(w)$, then color v by c. Otherwise choose a non colored vertex $v \in U$ arbitrarily and color it by the first available color.

2. If, after coloring some vertex in U, there is a vertex x_i such that at least $2m-1$ colors that are not included in $L(x_i)$ are used for coloring U, then delete x_i.

3. If, after coloring some vertex in U, there is a vertex $C_j^{(s)}$ with $s \in \{1, 2\}$ such that at least $n-2$ colors that are not included in $L(C_j^{(s)})$ are used for coloring U, then delete $C_j^{(s)}$.

This coloring of U can be constructed due the property that for each $v \in U$, $|L(v)| = k$ and $|U| = k - 2n < k$. Rule 2 is correct since $\deg_G(x_i) = k + 2m - 2$, and therefore if at least $2m - 1$ colors that are not included in $L(x_i)$ are used for coloring U, then any extension of the coloring of U to the coloring of $G - x_i$ can be further extended to the coloring of G, since there is at least one color in $L(x_i)$ which is not used for the coloring of neighborhood of this vertex. By same arguments, we can show the correctness of Rule 3 using the fact that $\deg_G(C_j^{(s)}) = k + n - 3$.

If after coloring the vertices of U, all vertices of $\{x_1, \ldots, x_n\} \cup \{C_1^{(1)}, \ldots, C_m^{(1)}\} \cup \{C_1^{(2)}, \ldots, C_m^{(2)}\}$ are deleted then we color remaining vertices $u_1, \ldots, u_{2n}$ greedily, and then we can claim that a list coloring of G exists by the correctness of Rules 2 and 3. Assume that at least one vertex of $\{x_1, \ldots, x_n\} \cup \{C_1^{(1)}, \ldots, C_m^{(1)}\} \cup \{C_1^{(2)}, \ldots, C_m^{(2)}\}$ was not deleted, and denote the set of such remaining vertices by W. Let $v \in U$ be the last colored vertex of U. Since $|U| = k - 2n = n + 4nm - 4m - 2n = n(2m - 1) + 2m(n - 2)$, the color list $L(v)$ contains at least $2n$ colors which are not used for coloring the vertices of U. Furthermore, for each $w \in W$, all these $2n$ colors are included in $L(w)$, due to the way we colored the vertices of U and since w was not deleted by Rules 2 or 3. We denote these unused colors by $1, \ldots, 2n$ and let $L = \{1, \ldots, 2n\}$. We proceed with coloring of G by coloring the vertices $u_1, \ldots, u_{2n}$ by the greedy algorithm using the first available color.

Assume without loss of generality that if some vertex u_i is colored by the color from L then it is colored by the color i. Now it remains to color the vertices of W. Notice that $G[W]$ is an induced subgraph of H. For each $w \in W$, denote by $L'(w)$ the colors from $L(w)$ which are not used for coloring vertices from the set $\{u_1, \dots, u_k\}$ that are adjacent to w. It can be easily seen that for any $x_i \in W$, $2i-1, 2i \in L'(x_1)$, for any $C_j^{(1)} \in W$ which corresponds to clause with literals x_p, x_q, x_r, $2p-1, 2q-1, 2r-1 \in L'(C_j^{(1)})$, and for any $C_j^{(2)} \in W$ which corresponds to clause with literals x_p, x_q, x_r, $2p, 2q, 2r \in L'(C_j^{(2)})$. Since there is a truth assignment for variables X such that each clause contains at least one true literal and at least one false literal, by Lemma 1 we can color the vertices of W.

To conclude the proof of the theorem, it remains to prove that G is P_5-free. Suppose that P is an induced path in G. Since H is a complete bipartite graph, P can contain at most 3 vertices of H and if it contains 3 vertices then these vertices have to be consecutive in P (notice that if P contains vertices only from one set of the bipartition of H, then the number of such vertices is at most 2 since they have to be joined by subpaths of P which go through vertices from the clique $\{u_1, \dots, u_k\}$). Also P can contain at most 2 vertices from the clique $\{u_1, \dots, u_k\}$, and if it has 2 vertices then they are consecutive. Hence, P has at most 5 vertices, and if P has 5 vertices then either $P = u_{t_1} u_{t_2} C_{j_1}^{(s_1)} x_i C_{j_2}^{(s_2)}$ or $P = u_{t_1} u_{t_2} x_{i_1} C_j^{(s)} x_{i_2}$. Assume that $P = u_{t_1} u_{t_2} C_{j_1}^{(s_1)} x_i C_{j_2}^{(s_2)}$. Since P is an induced path, vertices u_{t_1}, u_{t_2} are not adjacent to x_i. By the construction of G, it means that $\{t_1, t_2\} = \{2i-1, 2i\}$. But then $C_{j_2}^{(s_2)}$ is adjacent either u_{t_1} or u_{t_2}. Suppose that $P = u_{t_1} u_{t_2} x_{i_1} C_j^{(s)} x_{i_2}$. Again by the construction of G, $\{t_1, t_2\} = \{2i_2 - 1, 2i_2\}$ and $C_j^{(s)}$ is adjacent to u_{t_1} or u_{t_2}. By these contradictions, P has at most 4 vertices. □

5 Conclusion and open problems

We proved that the k-Choosability problem is FPT for P_5-free graphs when parameterized by k. It can be noted that our algorithm described in the proof of Theorem 1 does not explicitly use the absence of induced paths P_5. It is based on the property that any induced subgraph of a P_5-free graph has a dominating set of bounded (by some function of k) size. It would be interesting to construct a more efficient algorithm for k-Choosability which actively exploits the fact that the input graph has no induced P_5.

Another interesting question is whether it is possible to extend our result for P_r-free graphs for some $r \geq 6$? Particularly, it is known [18] that any P_6-free graph contains either a dominating biclique or a dominating induced cycle C_6. Is it possible to prove that k-Choosability is FPT for P_6-free graphs using this fact?

Also, we proved that k-Choosability is NP-hard for P_5-free graphs. Is this problem Π_2^P-complete?

Finally, what can be said about P_4-free graphs or *cographs*? It is possible to construct a more efficient algorithm using same ideas as in the proof of Theorem 1 and the well known fact (see e.g. [3]) that any cographs can be constructed from from isolated vertices by *disjoint union* and *join* operations, and such decomposition of any cograph can be constructed in linear time [4]? Instead of the presence of a dominating clique or a dominating P_3 we can use the property [13] that $ch(K_{r,r^r}) > r$. Unfortunately this algorithm is still double exponential in k. Is it possible to construct a better algorithm?

References

[1] N. Alon, *Restricted colorings of graphs*, in Surveys in combinatorics, 1993 (Keele), vol. 187 of London Math. Soc. Lecture Note Ser., Cambridge Univ. Press, 1993, pp. 1–33.

[2] G. Bacsó and Z. Tuza, *Dominating cliques in P_5-free graphs*, Period. Math. Hungar., 21 (1990), pp. 303–308.

[3] A. Brandstädt, V. B. Le, and J. P. Spinrad, *Graph classes: a survey*, SIAM Monographs on Discrete Mathematics and Applications, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1999.

[4] D. G. Corneil, Y. Perl, and L. K. Stewart, *A linear recognition algorithm for cographs*, SIAM J. Comput., 14 (1985), pp. 926–934.

[5] R. G. Downey and M. R. Fellows, *Parameterized complexity*, Monographs in Computer Science, Springer-Verlag, 1999.

[6] P. Erdős, A. L. Rubin, and H. Taylor, *Choosability in graphs*, in Proceedings of the West Coast Conference on Combinatorics, Graph Theory and Computing (Humboldt State Univ., Arcata, Calif., 1979), Congress. Numer., XXVI, Winnipeg, Man., 1980, Utilitas Math., pp. 125–157.

[7] J. Flum and M. Grohe, *Parameterized Complexity Theory*, Springer-Verlag, 2006.

[8] M. R. Garey and D. S. Johnson, *Computers and intractability*, W. H. Freeman and Co., San Francisco, Calif., 1979.

[9] S. Gutner and M. Tarsi, *Some results on (a:b)-choosability*, CoRR, abs/0802.1338 (2008).

[10] C. T. Hoàng, M. Kaminski, V. V. Lozin, J. Sawada, and X. Shu, *A note on k-colorability of p5-free graphs*, in MFCS, E. Ochmanski and J. Tyszkiewicz, eds., vol. 5162 of Lecture Notes in Computer Science, Springer, 2008, pp. 387–394.

[11] T. R. Jensen and B. Toft, *Graph Coloring Problems*, Wiley Interscience, 1995.

[12] D. Král, J. Kratochvíl, Z. Tuza, and G. J. Woeginger, *Complexity of coloring graphs without forbidden induced subgraphs*, in WG, A. Brandstädt and V. B. Le, eds., vol. 2204 of Lecture Notes in Computer Science, Springer, 2001, pp. 254–262.

[13] N. V. R. Mahadev, F. S. Roberts, and P. Santhanakrishnan, *3-choosable complete bipartite graphs*, Technical Report 49-91, Rutgers University, New Brunswick, NJ, 1991.

[14] R. Niedermeier, *Invitation to Fixed-Parameter Algorithms*, Oxford University Press, 2006.

[15] B. Randerath and I. Schiermeyer, *3-colorability $\in P$ for P_6-free graphs*, Discrete Appl. Math., 136 (2004), pp. 299–313. The 1st Cologne-Twente Workshop on Graphs and Combinatorial Optimization (CTW 2001).

[16] C. Thomassen, *Every planar graph is* 5*-choosable*, J. Combin. Theory Ser. B, 62 (1994), pp. 180–181.

[17] Z. Tuza, *Graph colorings with local constraints—a survey*, Discuss. Math. Graph Theory, 17 (1997), pp. 161–228.

[18] P. van 't Hof and D. Paulusma, *A new characterization of p6-free graphs*, in COCOON, X. Hu and J. Wang, eds., vol. 5092 of Lecture Notes in Computer Science, Springer, 2008, pp. 415–424.

[19] V. G. Vizing, *Coloring the vertices of a graph in prescribed colors*, Diskret. Analiz, (1976), pp. 3–10, 101.

[20] G. J. Woeginger and J. Sgall, *The complexity of coloring graphs without long induced paths*, Acta Cybernet., 15 (2001), pp. 107–117.