On the Tractability of (k, i)-Coloring[☆]

Sriram Bhyravarapu[a], Saurabh Joshi[a], Subrahmanyam Kalyanasundaram[a], Anjeneya Swami Kare[a,b]

[a] *Department of Computer Science and Engineering, IIT Hyderabad, Telangana - 502 285, India.*
[b] *School of Computer and Information Sciences, University of Hyderabad, Telangana - 500 046, India.*

Abstract

In an undirected graph, a proper (k, i)-coloring is an assignment of a set of k colors to each vertex such that any two adjacent vertices have at most i common colors. The (k, i)-coloring problem is to compute the minimum number of colors required for a proper (k, i)-coloring. This is a generalization of the classical graph coloring problem.

We design a parameterized algorithm for the (k, i)-coloring problem with the size of the feedback vertex set as a parameter. Our algorithm does not use tree-width machinery, thus answering a question of Majumdar, Neogi, Raman and Tale [CALDAM 2017]. We also give a faster exact algorithm for $(k, k-1)$-coloring. From the hardness perspective, we show that the (k, i)-coloring problem is NP-complete for any fixed values i, k, whenever $i < k$, thereby settling a conjecture of Méndez-Díaz and Zabala [1999] and again asked by Majumdar, Neogi, Raman and Tale. The NP-completeness result improves the partial NP-completeness shown in the preliminary version of this paper published in CALDAM 2018.

1. Introduction

In an undirected graph $G = (V, E)$, a *proper vertex coloring* is an assignment of colors to the vertices of the graph such that adjacent vertices get different colors. The classical graph coloring problem asks to compute the *chromatic number* of the graph (denoted by $\chi(G)$), which is the minimum number of colors required to properly color the graph. This is a well known NP-hard problem and has been studied in multiple directions.

[☆] This paper is the full version of the article "On the Tractability of (k, i)-Coloring"[1], published in the CALDAM 2018 conference, with the same set of authors.

Email addresses: cs16resch11001@iith.ac.in (Sriram Bhyravarapu), sbjoshi@iith.ac.in (Saurabh Joshi), subruk@iith.ac.in (Subrahmanyam Kalyanasundaram), askcs@uohyd.ac.in (Anjeneya Swami Kare)

Many variants and generalizations of the graph coloring problem have been studied in the past. In this paper we address a generalization of the graph coloring problem called (k, i)-coloring problem. For a proper (k, i)-coloring, we need to assign a set of k colors to each vertex such that the adjacent vertices share at most i colors. The (k, i)-coloring problem asks to compute the minimum number of colors required to properly (k, i)-color the graph. The minimum number of colors required is called the (k, i)-*chromatic number*, denoted by $\chi_k^i(G)$. Note that $(1, 0)$-coloring is the same as the classical graph coloring problem.

(k, i)-Coloring Problem
Instance: An undirected graph $G = (V, E)$.
Output: The (k, i)-chromatic number of G, $\chi_k^i(G)$.

We also define below the (q, k, i)-coloring problem, the decision version of the (k, i)-coloring problem.

(q, k, i)-Coloring Problem
Instance: An undirected graph $G = (V, E)$.
Question: Does G have a proper (k, i)-coloring using at most q colors?

The (k, i)-coloring problem was first studied by Méndez-Díaz and Zabala in [2]. For arbitrary k and i, the (k, i)-coloring problem is NP-hard because $(1, 0)$-coloring is NP-hard. Apart from studying the basic properties, they also gave an integer linear programming formulation of the problem. Stahl [3] and independently Bollobás and Thomason [4] introduced the $(k, 0)$-coloring problem under the names of k-tuple coloring and k-set coloring respectively. The k-tuple coloring problem has been studied in detail [5, 6], and Irving [7] showed that this problem is NP-hard as well. Some of the applications for the $(k, 0)$-coloring problem include construction of pseudorandom number generators, randomness extractors, secure password management schemes, aircraft scheduling, biprocessor tasks and frequency assignment to radio stations [8, 9]. Brigham and Dutton [10] studied another variant of the problem, where k colors have to be assigned to each vertex such that the adjacent vertices share exactly i colors.

Bonomo, Durán, Koch and Valencia-Pabon [11] studied the connection between the (k, i)-coloring problem on cliques and the theory of error correcting codes. In coding theory, a (j, d, k)-constant weight code represents a set of codewords of length j with exactly k ones in each codeword, with Hamming distance at least d. They observed a direct connection between $A(j, d, k)$, the largest possible size of a (j, d, k)-constant weight code, and the (k, i)-colorability of cliques and used the existing results from coding theory (such as the Johnson bound [12]) to infer results on the (k, i)-colorability of cliques. Finding bounds on $A(j, d, k)$ is a well-studied problem in coding theory, and lots of questions on $A(j, d, k)$ are still open. This indicates the difficulty of the (k, i)-coloring problem even on graphs as simple as cliques.

Since the (k, i)-coloring problem is NP-hard in general, it is natural to study the tractability for special classes of graphs. Polynomial time algorithms are only known for a few of such classes namely bipartite graphs, cycles, cacti and graphs with bounded vertex cover or tree-width [11, 13]. From the NP-hardness perspective, it is interesting to ask if the (k, i)-coloring problem is NP-hard for specific values of i. Except for the cases $i = k$, where the problem is trivial, and $i = 0$, where the problem is NP-hard [7], the NP-hardness remains open for all other values of i.

Recently, Majumdar, Neogi, Raman and Tale [13] studied the (k, i)-coloring problem and gave exact and parameterized algorithms for the problem. They showed that the problem is fixed parameter tractable (FPT) when parameterized by tree-width. As the tree-width is at most $(|S| + 1)$, where S is a feedback vertex set (FVS) of the graph, their algorithm also implies that (k, i)-coloring is FPT when parameterized by the size of FVS. As an open question, they asked to devise an FPT algorithm parameterized by the size of FVS, without going through tree-width.

We summarize our results below. In the following, n denotes the number of vertices of the graph.

- An algorithm for the (q, k, i)-coloring problem that runs in $O(\binom{q}{k}^{|S|+2} n^{O(1)})$ time and $O(\binom{q}{k} n)$ space and does not use tree-width machinery, where S is an FVS of the graph. This implies an FPT algorithm for the (q, k, i)-coloring problem parameterized by the size of FVS, and thus answers the question posed in [13].

  The algorithm in [13] runs in $O(\binom{q}{k}^{tw+1} n^{O(1)})$ time. Note that there is a polynomial time algorithm that approximates the FVS within a factor 2 [14], but there is no known polynomial time algorithm that approximates tree-width within a constant factor [15]. Moreover, computing the size of the smallest FVS is also known to be FPT parameterized by $|S|$. There has been a series of results improving the running time, the fastest known algorithm [16] runs in $O(3.619^{|S|} n^{O(1)})$ time.

  Building on our algorithm for the (q, k, i)-coloring problem, we also give an algorithm that computes the number of proper (q, k, i)-colorings of G, in $O(\binom{q}{k}^{|S|+2} n^{O(1)})$ time and $O(\binom{q}{k} n^2 \log q)$ space.

- We show that the decision version of the (k, i)-coloring problem is NP-complete for any fixed values k, i whenever $i < k$. This answers questions posed in [2] and [13] and settles the complexity of the (k, i)-coloring problem for all values of k and i. This result also improves the partial NP-completeness shown in the preliminary version [1] of this paper.

- We give a $2^n n^{O(1)}$ time exact algorithm for the $(k, k-1)$-coloring problem. This is a direct improvement over the $4^n n^{O(1)}$ time algorithm given in [13] for the same problem.

2. Preliminaries

A *parameterized problem* is a language $B \subseteq \Sigma^* \times \mathbb{N}$ where Σ is a fixed, finite alphabet. For example $(x, \ell) \in \Sigma^* \times \mathbb{N}$, here ℓ is called the parameter. A parameterized problem $B \subseteq \Sigma^* \times \mathbb{N}$ is called *fixed-parameter tractable* (FPT) if there is an algorithm $\mathcal{A}$, a computable function $f : \mathbb{N} \to \mathbb{N}$, and a constant c such that, given $(x, \ell) \in \Sigma^* \times \mathbb{N}$, the algorithm $\mathcal{A}$ correctly decides whether $(x, \ell) \in B$ in time bounded by $f(\ell)|x|^c$.

We assume that the graph is simple and undirected. We use n to denote $|V|$, the number of vertices of the graph. We say that the vertices u and v are *adjacent* (*neighbors*) if $\{u, v\} \in E$. For $v \in V$, we let $N(v)$ denote the set of neighbors of v. For $S \subseteq V$, the subgraph induced by S is denoted by $G[S]$. We use $O^*(f(n))$ to denote $O(f(n)n^{O(1)})$. We use the set of natural numbers for coloring the graph. We use the standard notations $[q] = \{1, 2, \ldots, q\}$ and $\binom{[q]}{k}$ to denote the set of all k-sized subsets of $[q]$. In the rest of the paper, we use the term *coloring* of a set $X \subseteq V$ to denote a mapping $h : X \to \binom{[q]}{k}$. We say that h is a *proper* (q, k, i)-coloring (or proper (k, i)-coloring) of X if any pair of adjacent vertices in X have no more than i colors in common. To avoid clutter, we resort to mild abuse of notation and use the phrase "the (k, i)-coloring problem is NP-complete" in place of "there exists a q for which the (q, k, i)-coloring problem is NP-complete".

3. (q, k, i)-Coloring Parameterized by Size of FVS

A *Feedback Vertex Set (FVS)* is a set of vertices $S \subseteq V$, removal of which from the graph G makes the remaining graph $(G[V \setminus S])$ acyclic. Many NP-hard problems have been shown to be tractable for graphs with bounded FVS [17]. In this section, we focus on the (q, k, i)-coloring problem for fixed values q, k, i. In [13], Majumdar, Neogi, Raman and Tale gave an $O(\binom{q}{k}^{tw+1} n^{O(1)})$ time[1] algorithm for the (q, k, i)-coloring problem, where tw denotes the tree-width of the graph. It is known that $tw \leq |S| + 1$ [18], where $|S|$ is the size of a smallest FVS of G. We present an algorithm for (q, k, i)-coloring that runs in $O(\binom{q}{k}^{|S|+2} n^{O(1)})$ time, that does not use the tree-width machinery.

A brief description of our algorithm follows. Let S be an FVS of G. We start with a coloring of the vertices of S. Recall that $G[V \setminus S]$ is a forest. Each of the connected components of $G[V \setminus S]$ is a tree. For each of these components, we traverse the tree bottom-up and use a dynamic programming technique to compute the list of all k-color sets that each vertex $w \in V \setminus S$ can take. For each $C \in \binom{[q]}{k}$, we include C in w's list if there is a coloring for the subtree rooted at w, consistent with the coloring of S, when w is given the color set C. We repeat this for all proper colorings of S.

[1] Even though [13] claims a running time of $O(\binom{q}{k}^{tw} n^{O(1)})$ for their algorithm, there is an additional factor of $\binom{q}{k}$ that is omitted, presumably because $\binom{q}{k}$ is treated as a constant.

Let $\Psi = \binom{[q]}{k}$ denote the family of all k-sized subsets of $[q]$. For any pair of sets $C, C' \in \Psi$, we say that (C, C') is *legal* if $|C \cap C'| \leq i$, and *illegal* if $|C \cap C'| > i$. Given two sets $C, C' \in \Psi$, it is easy to check if (C, C') is a legal pair. Formally, we have:

Proposition 1. *Given $C, C' \in \Psi$, it takes $O(k \log k)$ time to check if (C, C') is a legal pair.*

Definition 2. Consider a partial coloring $h : S \to \Psi$ where only the vertices of the FVS S are colored. For a vertex $w \in V \backslash S$ and a set $C \in \Psi$, we say that (w, C) is *h-compatible* if for all $x \in S \cap N(w)$, the pair $(C, h(x))$ is legal.

The set $\{C \in \Psi \mid (w, C)$ is h-compatible$\}$ is defined to be the set of *h-compatible colorings of w*.

Proposition 3. *Let $h : S \to \Psi$ be a coloring of the vertices in S. Let $w \in V \backslash S$ and $d_S(w) = |N(w) \cap S|$. Then the set of h-compatible colorings of w can be computed in time $O\left(\binom{q}{k} d_S(w) k \log k\right)$.*

Proof. For each $C \in \Psi$, we check if (w, C) is h-compatible. For this, we need to check for all neighbors x of w in S, whether $(C, h(x))$ is legal. The total running time is $\binom{q}{k} \cdot d_S(w) \cdot O(k \log k)$. □

Definition 4. Given a graph $G = (V, E)$ and a coloring $h : X \to \Psi$ for some $X \subseteq V$, we say that the coloring $h' : V \to \Psi$ is an *extension of h*, or *extends h* if for all $v \in X$, we have $h(v) = h'(v)$.

Lemma 5. *Given a proper (q, k, i)-coloring h of the vertices in a feedback vertex set S of the graph $G = (V, E)$, we can determine if h can be extended to a proper (q, k, i)-coloring of V in $O(\binom{q}{k}^2 n^{O(1)})$ time.*

Proof. The graph $G[V \backslash S]$ is a forest because S is a feedback vertex set. Therefore each connected component of $G[V \backslash S]$ is a tree. Below, we describe an algorithm that we can apply to each of these trees to yield a proper (q, k, i)-coloring extending h for the trees. Combining the colorings, we get a proper (q, k, i)-coloring of V, that is an extension of h.

Let T denote one of the trees in the forest. We designate an arbitrary vertex r of T as its root. Let T_w denote the subtree rooted at a node $w \in T$.

We maintain a table with $|V(T)|$ rows, one for each vertex $w \in V(T)$, and $\binom{q}{k}$ columns, one for each k-set $C \in \Psi$. The entry at row w and column C is denoted by $M_w(C)$. The 0/1 entry $M_w(C)$ indicates whether there is a proper (q, k, i)-coloring of T_w, with w assigned the set C, consistent with the coloring h of S.

We will process T in a post order fashion as follows:

1. **When w is a leaf in T:** In this case, we set $M_w(C) = 1$ if (w, C) is h-compatible. Otherwise, we set $M_w(C) = 0$.
   For any leaf w, the values $M_w(C)$ corresponding to all $C \in \Psi$ can be computed in time $O(\binom{q}{k} d_S(w) k \log k)$ by Proposition 3. Here $d_S(w)$ denotes the number of neighbors of w in S.

2. **When w is an internal node in T:** Let $u_1, u_2, \ldots, u_\ell$ be the children of w in T. Recall that we process T in post order fashion. Before we process w, the M_{u_j} values for all the children of w would already have been computed. The value $M_w(C)$ is computed as follows:
   - If (w, C) is not h-compatible, we set $M_w(C) = 0$.
   - If (w, C) is h-compatible, we do the following:
     - If for each child u_j of w, there exists at least one coloring $C' \in \Psi$ such that $M_{u_j}(C') = 1$ and (C, C') is a legal pair, then set $M_w(C) = 1$.
     - Otherwise set $M_w(C) = 0$.

   For each w and C, the h-compatibility check takes $O(d_S(w)k \log k)$ time. If (w, C) is h-compatible, we need to check all the children u_j, and the table entries $M_{u_j}(C')$ for all $C' \in \Psi$. Together with the check for (C, C') being a legal pair, the computation takes $d_T(w) \cdot \binom{q}{k} \cdot O(k \log k)$ time, where $d_T(w)$ is the number of children of w in the tree T.
   Adding all up, the computation of the table entries for w takes time

$$O\left(\binom{q}{k} \cdot k \log k \cdot \left[d_S(w) + d_T(w)\binom{q}{k}\right]\right). \tag{1}$$

If for some $C \in \Psi$, $M_r(C) = 1$, then we know that there exists a proper (q, k, i)-coloring of T that is consistent with the coloring h of S.

The time complexity is obtained by adding the expression in (1) over all the vertices $w \in V \backslash S$. Using the bounds $d_S(w) \leq n$ and $\sum_{\text{Trees } T} \sum_{w \in V(T)} d_T(w) \leq \sum_{\text{Trees } T} |V(T)| \leq n$, we get that the time complexity is upper bounded by

$$O\left(\binom{q}{k} \cdot k \log k \cdot \left[n^2 + n\binom{q}{k}\right]\right),$$

which is at most $O\left(\binom{q}{k}^2 \cdot n^2\right)$, by noting that k is a constant. □

The correctness of the procedure explained in the above lemma can be proved using an induction on the vertices of T according to its post order traversal. The inductive claim says that $M_w(C) = 1$ if and only if there is a proper (q, k, i)-coloring of T_w, with w assigned the set C, consistent with the given coloring of S.

Lemma 6. *Given a proper (q, k, i)-coloring h of the vertices in a feedback vertex set S of the graph $G = (V, E)$, we can determine if h can be extended to a proper (q, k, i)-coloring of V with space complexity $O(\binom{q}{k}n)$.*

Proof. Recall the algorithm explained in Lemma 5. At each vertex w in $G[V \setminus S]$, we need $O(\binom{q}{k})$ space to store values $M_w(C)$ for all $C \in \Psi$. □

Theorem 7. *The (q, k, i)-coloring problem can be solved in time $O((\binom{q}{k})^{|S|+2} n^{O(1)})$ and $O(\binom{q}{k}n)$ space, where S is a feedback vertex set of G.*

PROOF. For each color assignment h of S, we first determine if h is a proper (q,k,i)-coloring. This can be done in $O(|S|^2 k \log k)$ time. Then we determine whether there exists a proper (q,k,i)-coloring that extends h in $O(\binom{q}{k}^2.n^{O(1)})$ time by Lemma 5. Since there are at most $\binom{q}{k}^{|S|}$ many colorings of S, we can determine whether there exists a proper (q,k,i)-coloring of G in $O(\binom{q}{k}^{|S|+2}n^{O(1)})$ time.

We need $O(|S|k\log q)$ space to store the coloring h of S. And by Lemma 6, we need $O(\binom{q}{k}n)$ space to determine if h can be extended to a proper coloring of G. The latter is the dominating term and determines the total space requirement of the algorithm. □

On generating a proper (q,k,i)-coloring. We observe that we can modify Theorem 7 to obtain an algorithm that generates a proper (q,k,i)-coloring of G, if one exists. After executing the steps of the algorithm corresponding to Theorem 7, we traverse the tree in top-down fashion from the root, and for each vertex $w \in T$, we find a color assignment that is consistent with the color set assigned to its parent, subtree T_w and coloring of S. The latter two are already encoded in $M_w(C)$ value. The asymptotic time and space complexity are the same as that in Theorem 7.

We would like to observe a difference in the space usage of our FPT algorithm to the FPT algorithm for (q,k,i)-coloring parameterized by tree-width in [13]. We note that the algorithm in [13] can also be modified similarly to obtain an algorithm that generates a proper coloring. However, such an algorithm would require to store all feasible colorings at each bag of the tree-decomposition, resulting in a $O(\binom{q}{k}^{tw+1})$ space usage at each bag. Since there are $O(n)$ bags, total space required by the algorithm is $O(\binom{q}{k}^{tw+1}n)$, which is significantly larger than the $O(\binom{q}{k}n)$ space required by our algorithm.

Decision vs. search problem. We now note that $\chi^i_k(G) \leq \chi^i_k(G[S]) + \chi^i_k(G[V\backslash S])$. In the RHS, the first term $\chi^i_k(G[S]) \leq k|S|$ trivially, and the second term $\chi^i_k(G[V\backslash S]) \leq 2k - i$ since $G[V\backslash S]$ is a forest. Thus we have $\chi^i_k(G) \leq k|S| + 2k - i \leq k(|S|+2)$.

We note that we could run the algorithm for (q,k,i)-coloring and perform binary search between 1 and $k(|S|+2)$ and determine $\chi^i_k(G)$, the smallest q for which the graph has a proper (q,k,i)-coloring. The running time of this procedure would be at most $\log(k(|S|+2))$ times the running time of the algorithm presented in Theorem 7, that is $\log(k(|S|+2)) \cdot O(\binom{k(|S|+2)}{k}^{|S|+2}n^{O(1)})$.

Thus the FPT algorithm parameterized by the size of the FVS for the (q,k,i)-coloring problem implies an FPT algorithm parameterized by the size of the FVS for the (k,i)-coloring problem as well.

Counting all proper (q,k,i)-colorings. Here we show that we can modify the algorithm described in Lemma 5 to count the number of proper (q,k,i)-colorings of G. Let a proper (q,k,i)-coloring h of FVS S be given. Instead of maintaining $M_w(C)$ for a vertex w in a rooted tree T, we maintain another value $M^{\#}_w(C)$.

$$M_w^{\#}(C) = \begin{cases} 0 & \text{if } (w, C) \text{ is not } h\text{-compatible.} \\ 1 & \begin{cases} \text{if } w \text{ is a leaf,} \\ \text{and } (w, C) \text{ is } h\text{-compatible.} \end{cases} \\ \displaystyle\prod_{\forall u_j \in \text{child}(w)} \sum_{\text{legal}(C, C')} M_{u_j}^{\#}(C') & \begin{cases} \text{if } w \text{ is a non-leaf vertex,} \\ \text{and } (w, C) \text{ is } h\text{-compatible.} \end{cases} \end{cases}$$

At each vertex w, $M_w^{\#}(C)$ maintains a count of the proper (q, k, i)-colorings of T_w, consistent with the coloring h of S, where w gets assigned the set C. The correctness can be verified by a straightforward induction on the tree vertices in post order traversal. If r is the root of T, $M_r^{\#}(C)$ gives the count of proper (q, k, i)-colorings of T, where r is colored C, consistent with the coloring h of S.

The total number of proper (q, k, i)-colorings of G is therefore computed by taking into account (i) all proper (q, k, i)-colorings h of S, (ii) all the trees T_j in $G[V \backslash S]$, and (iii) all color sets $C \in \Psi$ at the root of T_j. The full expression is as follows:

$$\text{No. of proper } (q, k, i)\text{-colorings } = \sum_{\substack{\text{proper } (q,k,i)\text{-} \\ \text{colorings of } S}} \left(\prod_{T_j \text{ in } G[V \backslash S]} \left(\sum_{C \in \Psi} M_{\text{root}(T_j)}^{\#}(C) \right) \right).$$

The above expression implies the following theorem. The asymptotic time complexity remains the same as Theorem 7, whereas the space complexity incurs a blowup of $nk \log q$, because of the maximum value $M_w^{\#}(C)$ can take.

Theorem 8. *There is an algorithm that computes the number of proper (q, k, i)-colorings of G, in $O(\binom{q}{k}^{|S|+2} n^{O(1)})$ time and $O(\binom{q}{k} n^2 \log q)$ space, where S is a feedback vertex set of G.*

4. Faster Exact Algorithm for $(k, k-1)$-coloring

The article [13] gave an $O^*(4^n)$ time exact algorithm for the $(k, k-1)$-coloring problem. Their algorithm was based on running an exact algorithm for a set cover instance where the universe is the set of all the vertices V and the family of sets $\mathcal{F}$ is the set of all independent sets of vertices of G. To show correctness and running time, they used a claim[2] that relates $\chi_k^{k-1}(G)$ to the size of solution of the set cover instance, an $O(2^n \cdot n \cdot |\mathcal{F}|)$ time exact algorithm for the set cover problem [19] and an upper bound of 2^n on the size of the family of sets $\mathcal{F}$. Hence, the time complexity of their algorithm is $O(2^n \cdot n \cdot 2^n) = O(4^n \cdot n)$.

[2]The claim states that $\chi_k^{k-1} = q$, where q is the smallest integer such that $r \leq \binom{q}{k}$, and r is the size of a minimum solution of the set cover instance $(V(G), \mathcal{F})$. This claim is unnumbered, and appears in page 287 of CALDAM 2017 proceedings, in which [13] was published.

We first note that their algorithm also works when $\mathcal{F}$ is replaced by $\mathcal{F}'$, the set of all maximal independent sets of G. This is because any independent set $A \in \mathcal{F}$ is contained in a maximal independent set $A' \in \mathcal{F}'$. In any set covering of V using elements of $\mathcal{F}$, each set A can be replaced by an $A' \in \mathcal{F}'$, thus obtaining a set cover of V using elements of only $\mathcal{F}'$. By using the $3^{n/3}$ upper bound of Moon and Moser [20] on the number of maximal independent sets, the time complexity improves to $O(2^n \cdot n \cdot 3^{n/3}) = O(2.88^n \cdot n)$.

We now present a faster $O^*(2^n)$ algorithm to determine $\chi_k^{k-1}(G)$. Assuming the exact coloring algorithm in [21], this also has a simple proof of correctness.

Lemma 9. *For any graph G, $\chi_k^{k-1}(G) = q$ where q is the smallest integer such that $\binom{q}{k} \geq \chi_1^0(G)$. Thus there is a polynomial time reduction from the $(k, k-1)$-coloring problem to the $(1, 0)$-coloring problem.*

PROOF. The $(k, k-1)$-coloring problem asks to assign sets of k colors to each vertex, with the requirement that neighboring vertices must have distinct sets assigned to them. We may view each of the k-sized subsets as a color, and the $(1, 0)$-chromatic number $\chi_1^0(G)$ is the number of distinct k-sized subsets required.

Thus $\chi_k^{k-1}(G)$ is the smallest q that will provide $\chi_1^0(G)$ number of k-sized subsets. The polynomial time reduction is immediate. □

Combining the above lemma with the $O^*(2^n)$ time algorithm of Koivisto [21] to compute $\chi_1^0(G)$, we get the following theorem.

Theorem 10. *There is an algorithm with $O^*(2^n)$ time complexity that computes the $(k, k-1)$-chromatic number of a given graph.*

Further, we can infer from Lemma 9 that for those graphs G where we can compute χ_1^0(G) in polynomial time, $\chi_k^{k-1}(G)$ can also be determined in polynomial time. For instance, $\chi_k^{k-1}(K_n)$ can be computed in polynomial time as $\chi_1^0(K_n) = n$.

5. NP-completeness results

Since the (k, i)-coloring problem is a generalization of the $(1, 0)$-coloring problem, it follows that (k, i)-coloring is NP-hard in general. Notice that we have $\chi_k^k(G) = k$ for all graphs G. Thus the (k, k)-coloring problem is trivial. Méndez-Díaz and Zabala [2] conjectured that the (k, i)-coloring problem is NP-hard whenever $i < k$. In this section, we prove their conjecture by showing that the (k, i)-coloring problem is NP-complete for all values of k and i, as long as $i < k$. Given a coloring, we can easily verify if it is a proper (k, i)-coloring in polynomial time. Therefore, we will only be proving the NP-hardness aspect of NP-completeness.

5.1. Simple proofs for $(k,1)$-coloring and $(k,k-1)$-coloring

In this section, we provide simple proofs for the NP-completeness of $(k,1)$-coloring and $(k,k-1)$-coloring. For the $(k,0)$-coloring problem, we have the following result by Irving.

Theorem 11 ($(k,0)$-coloring is NP-complete [7]). *The $(2k+1,k,0)$-coloring problem is NP-complete for all $k \geq 1$.*

The NP-completeness of the $(k,k-1)$-coloring problem is claimed by [2]. However, we are unable to follow and verify the proof. We provide an alternate NP-hardness proof as a consequence of the correspondence in Lemma 9.

Theorem 12. *The $(k,k-1)$-coloring problem is NP-complete for all $k \geq 1$.*

PROOF. We use reductions from the $(1,0)$-coloring problem. We show that the $(q,k,k-1)$-coloring problem is NP-complete for all values of $q > k \geq 2$. From the correspondence in Lemma 9, it follows that for any given $k \geq 1$, a graph G is $(q,k,k-1)$-colorable if and only if G is $(\binom{q}{k},1,0)$-colorable. Since the $(r,1,0)$-coloring problems are NP-complete for all $r \geq 3$, it follows that $(\binom{q}{k},1,0)$-coloring problems are NP-complete for all $q > k \geq 2$, and hence we get that the $(q,k,k-1)$-coloring problems are NP-complete for all $q > k \geq 2$. □

The following lemmas will help us in proving further NP-completeness results.

Lemma 13 (Complement trick). *For integers $k, i \geq 1$, any graph G is $(2k+i,k+i,i)$-colorable if and only if it is $(2k+i,k,0)$-colorable.*

PROOF. Let $f : V \to \binom{[2k+i]}{k}$ be a $(2k+i,k,0)$-coloring of G. Consider the coloring f' where each vertex v is assigned the complement set $[2k+i] \backslash f(v)$. Notice that, every vertex is assigned $(k+i)$ colors, and any pair of adjacent vertices will share exactly i colors in the coloring f'. Thus we have a $(2k+i,k+i,i)$-coloring of G.

Similarly, if we start from a $(2k+i,k+i,i)$-coloring of G, we can get to a $(2k+i,k,0)$-coloring by taking the complement coloring. □

Theorem 11 and the above lemma together imply the NP-completeness of $(k,1)$-coloring for all $k \geq 2$.

Theorem 14 ($(k,1)$-coloring is NP-complete). *The $(2k+1,k+1,1)$-coloring problem is NP-complete for all $k \geq 1$.*

5.2. NP completeness of (k, i)-coloring

In this section, we prove the NP-completeness of (k, i)-coloring. The main ingredient of the NP-completeness result is the following theorem, which generalizes Theorem 11.

Theorem 15. *The $(2k+i, k, 0)$-coloring problem is NP-complete for all $k, i \geq 1$.*

Most of the remaining part of this section will be used to prove Theorem 15. We first show how to infer the NP-completeness of (k, i)-coloring from Theorem 15. The result is formally stated below:

Theorem 16 ((k, i)-coloring is NP-complete). *The (k, i)-coloring problem is NP-complete when $i < k$. That is, for any fixed values i, k such that $i < k$, there exists a q such that the (q, k, i)-coloring problem is NP-complete.*

Proof. Let k', i be such that $k' > i \geq 0$. We will show that (k', i)-coloring is NP-complete. Theorem 15 shows the NP-completeness of $(k', 0)$-coloring when $k' \geq 1$.

To show the NP-completeness of (k', i)-coloring when $k' > i \geq 1$, let $k = k' - i$. Notice that $k, i \geq 1$. By Theorem 15, we have that $(2k + i, k, 0)$-coloring is NP-complete. By Lemma 13 (complement trick), a graph is $(2k + i, k, 0)$-colorable if and only if it is $(2k+i, k+i, i)$-colorable. Hence the $(2k+i, k+i, i)$-coloring problem is NP-complete as well. Substituting $k' = k + i$, we get that the $(2k' - i, k', i)$-coloring problem is NP-complete. □

Now we shall start working towards the proof of Theorem 15. The *Kneser graph* forms an important component of the proof.

Definition 17 (Kneser Graph). The Kneser graph $K(r, k)$ is the graph whose vertices are $\binom{[r]}{k}$, the k-sized subsets of $[r]$, and vertices x and y are adjacent if and only if $x \cap y = \emptyset$ (when x and y are viewed as sets).

Consider the Kneser graph $K(r, k)$. We may view the elements of the set $[r]$ as colors. In this case, the sets associated with the vertices themselves form a proper $(k, 0)$-coloring of $K(r, k)$. We will call this $(k, 0)$-coloring as the *natural coloring*, denoted by h_N. This is because two vertices are adjacent if and only if they do not share any colors.

The following theorem states that when $r \geq 2k + 1$, the natural coloring is essentially the only proper $(k, 0)$-coloring of the Kneser graph $K(r, k)$ that uses r colors.

Theorem 18. *Let $r \geq 2k + 1$ and $k \geq 1$. Any $(r, k, 0)$-coloring h of $K(r, k)$ can be obtained from a natural coloring h_N by a permutation of colors.*

Before proving the above theorem, we state two known results that would be necessary in the proof.

Theorem 19 (Erdős-Ko-Rado Theorem [22]). *Let $r \geq 2k$. The largest independent set of the Kneser graph $K(r, k)$ is of size $\binom{r-1}{k-1}$.*

Theorem 20 (Hilton and Milner [23]). *Let $r > 2k$. Every independent set of the Kneser graph $K(r,k)$ of size $\binom{r-1}{k-1}$ are the set of vertices that contain some color a in their natural coloring.*

We first require the following lemma.

Lemma 21. *Let $r \geq 2k$ and $k \geq 1$. Any $(k,0)$-coloring of $K(r,k)$ requires at least r colors. If h is a $(r,k,0)$-coloring of $K(r,k)$, then each color in h must occur exactly $\binom{r-1}{k-1}$ times.*

Proof. Let h_N be the natural coloring and h be an arbitrary $(r,k,0)$-coloring of $K(r,k)$. Each color occurs exactly $\binom{r-1}{k-1}$ times in h_N. Below, we argue that this must happen in h as well.

All the vertices containing a specific color form an independent set. If any color in h occurs more than $\binom{r-1}{k-1}$ times, then we will have a contradiction to Theorem 19. Now notice that the total number of "color slots" is fixed at $k \cdot \binom{r}{k}$. Hence each color must occur $k \cdot \binom{r}{k}/r = \binom{r-1}{k-1}$ times on average. If any color in h occurs strictly less than $\binom{r-1}{k-1}$ times, then some other color has to occur more than $\binom{r-1}{k-1}$ times in order to compensate. Hence each color in h must occur exactly $\binom{r-1}{k-1}$ times.

Similarly, a $(k,0)$-coloring of $K(r,k)$ using less than r colors also implies an independent set larger than $\binom{r-1}{k-1}$, contradicting Theorem 19. □

Now we prove Theorem 18.

Proof (Proof of Theorem 18). Let $r \geq 2k+1$, $k \geq 1$ and $G = K(r,k)$. Without loss of generality, let the colors used in h be from the set $[r]$. We will show that there is a permutation function $\sigma : [r] \to [r]$ such that every vertex v with $h(v) = \{c_1, c_2, \ldots, c_k\} \subseteq [r]$, has natural coloring $h_N(v) = \{\sigma(c_1), \sigma(c_2), \ldots, \sigma(c_k)\}$.

By Lemma 21, each color in h must occur exactly $\binom{r-1}{k-1}$ times in G. For any color $c \in [r]$, the vertices with color c in h form an independent set of size $\binom{r-1}{k-1}$. Now Theorem 20 states that every independent set of size $\binom{r-1}{k-1}$ is the set of vertices that contain some color a in their natural coloring. Setting $a = \sigma(c)$, we get the desired mapping between the colors in h_N and h.

To complete the proof, we need to show that the mapping σ is a permutation. Since both domain and codomain of σ is $[r]$, it is enough to show that σ is injective. Assume for the sake of contradiction that there are two colors $c_1 \neq c_2$ such that $\sigma(c_1) = \sigma(c_2) = a$. Let X be the set of vertices in G that contain the color a in the natural coloring h_N. By our argument, note that X is also the set of vertices in G that contain the color c_1 in h (and also the vertices that contain the color c_2 in h).

Remove the set of vertices X from G. What remains is a copy of the Kneser graph $K(r-1,k)$, which is $(k,0)$-colored by h using $r-2$ colors. Since $r-1 \geq 2k$, by Lemma 21, the graph $K(r-1,k)$ cannot be $(k,0)$-colored by $r-2$ colors. This contradicts the assumption that there are colors $c_1 \neq c_2$ such that $\sigma(c_1) = \sigma(c_2) = a$. Hence σ is injective and a permutation, completing the proof. □

We conclude this section with one more definition, that of a totally independent 3-set in a Kneser graph.

Definition 22 (Totally Independent 3-Set). For $r \geq 2k+1$, consider the Kneser graph $K(r,k)$. Any set of three vertices of $K(r,k)$ form a *totally independent 3-set* if they all share the set of same $k-1$ colors in their natural coloring, and differ only in the last color.

Because of Theorem 18, notice that the above definition remains unchanged if we use any $(r,k,0)$-coloring instead of the natural coloring.

5.3. Proof of Theorem 15

In this subsection, we prove Theorem 15 which states that the $(2k+i,k,0)$-coloring problem is NP-complete for all $k,i \geq 1$.

As before, it is easy to see that the problem is in NP and we shall focus on showing that the problem is NP-hard. We will show that there is a polynomial time reduction from 3-*CNFSAT* to $(2k+i,k,0)$-coloring. Given a 3-CNF Boolean formula ψ, we will construct a graph G in polynomial time such that G is $(2k+i,k,0)$-colorable if and only if ψ is satisfiable. The formula ψ is a conjunction of clauses, with each clause consisting of exactly 3 literals. For a fixed value of k and i, we now describe the construction of the graph G from ψ.

Construction of G from ψ: Let ψ be a 3-CNF formula with n variables and m clauses. We first describe the vertices of the graph G.

V (i) Two vertices u and w.

V (ii) A set of $\binom{2k+i-2}{k} + \binom{2k+i-2}{k-2}$ vertices denoted as follows:

$$A = \left\{ v_\ell \mid \ell = 1,2,3,\ldots, \left[\binom{2k+i-2}{k} + \binom{2k+i-2}{k-2} \right] \right\}.$$

V (iii) For each variable p in ψ, a set B_p of $2 \cdot \binom{2k+i-2}{k-1}$ vertices. Each set B_p is defined as follows:

$$B_p = \{x_p, \overline{x_p}\} \cup \left\{ y_{p,\ell} \mid \ell = 1,2,3,\ldots, \left[2\binom{2k+i-2}{k-1} - 2 \right] \right\}.$$

V (iv) For each clause C_j in ψ, a set of three vertices denoted by $z_{j,1}, z_{j,2}$, and $z_{j,3}$.

V (v) For each clause C_j in ψ, a set of $\binom{2k+i}{k}$ vertices denoted by Γ_j defined as follows:

$$\Gamma_j = \left\{ \gamma_{j,\ell} \mid \ell = 1,2,3,\ldots, \binom{2k+i}{k} \right\}.$$

Thus the total number of vertices in G is $2 + \binom{2k+i-2}{k} + \binom{2k+i-2}{k-2} + 2n\binom{2k+i-2}{k-1} + m\left(3 + \binom{2k+i}{k}\right)$. For fixed values of k and i, the number of vertices in G is polynomial in the size of the formula ψ.

Now we describe the edges of G. While describing the edges, we use the shorthand notation uw for the edge $\{u,w\}$.

E (i) For each $1 \leq p \leq n$, the vertices $A \cup B_p$ form[3] a copy of the Kneser graph $K(2k+i,k)$. When the vertices of $K(2k+i,k)$ are regarded as the k-sized subsets of $[2k+i]$, the vertices in A correspond to the k-sized subsets of $[2k+i] \setminus \{2k, 2k+1\}$ and the $(k-2)$-sized subsets of $[2k+i] \setminus \{2k, 2k+1\}$ union with $\{2k, 2k+1\}$. That is, vertices in A correspond to the k-sized subsets of $[2k+i]$ that either contain both $2k$ and $2k+1$, or contain none of $2k$ and $2k+1$.
The vertices x_p and $\overline{x_p}$ correspond to the sets $[k-1] \cup \{2k\}$ and $[k-1] \cup \{2k+1\}$ in some order. The vertices in B_p correspond to the other k-sized subsets of $[2k+i]$, that contain either $2k$ or $2k+1$ but not both.

E (ii) Let $u', w' \in A$ be the vertices such that u' represents the set $\{1, 2, \ldots, k\}$ and w' represents the set $\{k, k+1, \ldots, 2k-1\}$. Then uw, uu' and ww' are edges.

E (iii) Let $U', W' \subseteq A$ defined as follows[4]. The set U' represents those set of vertices which have the set $\{1, 2, \ldots, k-1\}$ with the k-th element from $\{2k+2, 2k+3, \ldots, 2k+i\}$ while W' represents those set of vertices which have the set $\{k, k+1, \ldots, 2k-2\}$ with the k-th element from $\{2k+2, 2k+3, \ldots, 2k+i\}$. Note that $|U'| = |W'| = i-1$.
Now join u to all the vertices in U' using edges, and similarly join w to all the vertices in W'. [5]

E (iv) For each $1 \leq j \leq m$, $wz_{j,1}, wz_{j,2}$ and $wz_{j,3}$ are edges.

E (v) Suppose the j-th clause is $C_j = x_{j,1} \vee x_{j,2} \vee x_{j,3}$, where each of $x_{j,1}$, $x_{j,2}$ and $x_{j,3}$ are literals x_p or $\overline{x_p}$ for some $1 \leq p \leq n$. Then $z_{j,1}\overline{x_{j,1}}$, $z_{j,2}\overline{x_{j,2}}$ and $z_{j,3}\overline{x_{j,3}}$ are edges, for each j.

E (vi) For each j, the vertex set Γ_j forms a copy of $K(2k+i,k)$.

E (vii) For each j, we identify three vertices $t_{j,1}$, $t_{j,2}$ and $t_{j,3}$ from Γ_j, such that these three vertices form a totally independent 3-set in Γ_j.
Then $z_{j,1}t_{j,1}$, $z_{j,2}t_{j,2}$ and $z_{j,3}t_{j,3}$ are edges for each j.

E (viii) For each j, all the vertices in $\{z_{j,1}, z_{j,2}, z_{j,3}\}$ are joined to all the vertices in U', forming a complete bipartite graph.

E (ix) Similarly, for each j, the sets $\{t_{j,1}, t_{j,2}, t_{j,3}\}$ and W' form a complete bipartite graph.

The description of the graph G is complete. A pictorial representation is given in Figure 1. We now claim that the graph G so constructed is $(2k+i, k, 0)$-colorable if and only if ψ is satisfiable.

Colorability implies satisfiability: Suppose that G is $(2k+i, k, 0)$-colorable. We will construct a satisfying assignment for ψ. Given a $(2k+i, k, 0)$-coloring

[3] Notice that $|A| + |B_p| = \binom{2k+i}{k}$.

[4] We remark that when $i = 1$, the sets U' and W' are empty sets. However, the correctness of proof is maintained even in this case.

[5] We can reduce the sizes of both U' and W' to $\lceil (i-1)/k \rceil$. This can be done by forming the k-sized sets from $\{2k+2, 2k+3, \ldots, 2k+i\}$ in increasing order. However, we use the simpler (but larger) sets U' and W' in the proof.

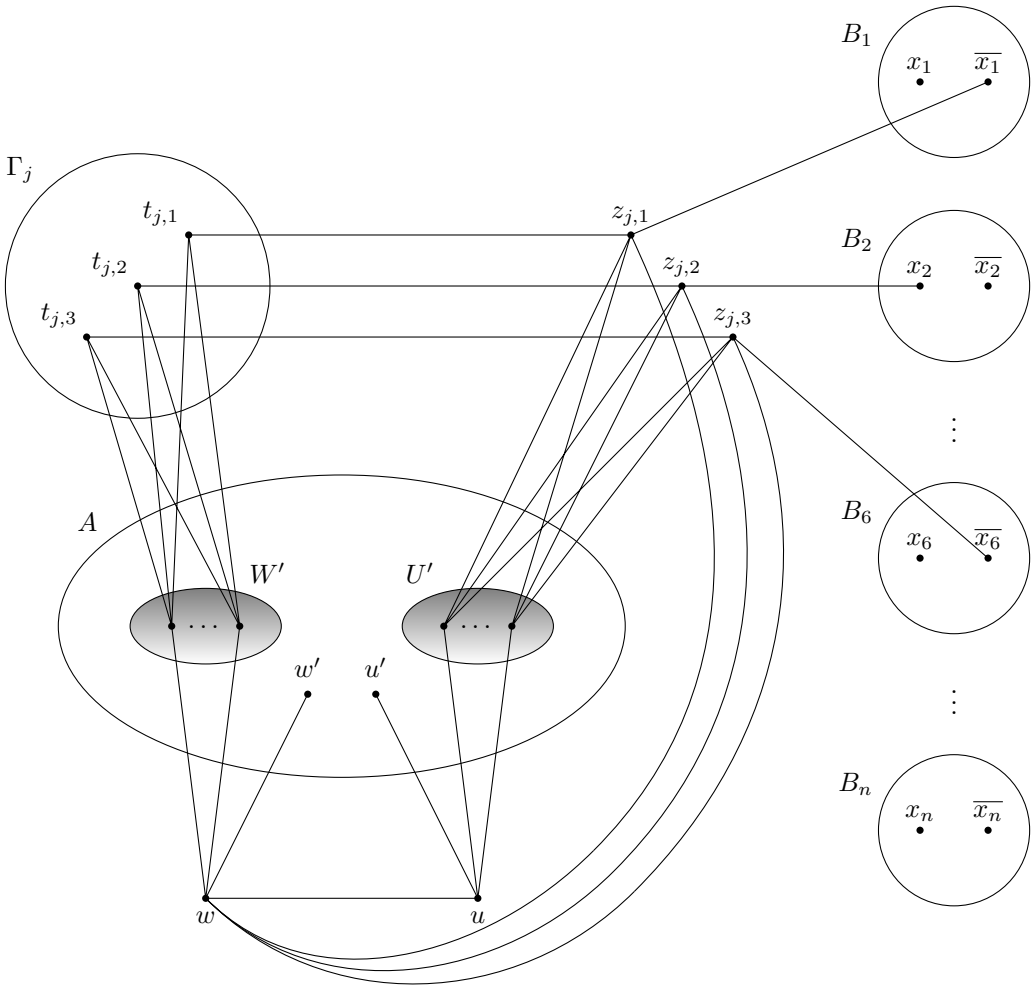

Figure 1: The graph G that corresponds to the formula ψ. The figure depicts the edges of the graph G for the clause $C_j = x_1 \vee \overline{x_2} \vee x_6$. For each clause in ψ, there will be a copy of vertices Γ_j, $z_{j,1}$, $z_{j,2}$ and $z_{j,3}$ and the corresponding edges. For reducing clutter, several vertices and edges have not been shown in the figure.

of G, we can assume that the vertices of the set A are colored according to the natural coloring of the Kneser graph $A \cup B_p$ for any p using Theorem 18. Using the vertices of A, the correspondence between the colors in the natural coloring and the given coloring is determined, except for the colors $2k$ and $2k+1$.

The vertex u is adjacent to u', which is colored $\{1, 2, \ldots, k\}$ and to the set U' whose vertices are colored with the colors $\{1, 2, \ldots, k-1\} \cup \{2k+2, 2k+3, \ldots, 2k+i\}$. This leaves the colors $\{k+1, k+2, \ldots, 2k, 2k+1\}$ available for coloring u. Similarly for w, because of adjacencies to w' and W', the available

colors are $\{1, 2, \ldots, k-1\} \cup \{2k, 2k+1\}$. Further, because uw is an edge, the colors $\{k+1, k+2, \ldots, 2k-1\}$ are fixed for u and $\{1, 2, \ldots, k-1\}$ for w, with u and w being assigned one color each from $\{2k, 2k+1\}$. Without loss of generality, we may assume that u gets the color $2k$ and w gets the color $2k+1$. We will further refer to $2k$ as the 'true color' and $2k+1$ as the 'false color'.

Given the way A is colored, there are two ways in which each B_p can be colored consistent with the coloring of A. In each B_p, the vertices x_p can be colored $\{1, 2, \ldots, k-1, 2k\}$ and $\overline{x_p}$ be colored $\{1, 2, \ldots, k-1, 2k+1\}$ or vice versa. The coloring of the rest of the vertices in B_p can be determined from the coloring of x_p and $\overline{x_p}$. The coloring of the vertices x_p and $\overline{x_p}$ in the given coloring of G will help us construct the satisfying assignment for the formula ψ. If the vertex x_p is colored $\{1, 2, \ldots, k-1, 2k\}$, then the literal x_p is considered to be a true literal (and $\overline{x_p}$ false) since it has been assigned the 'true color' $2k$. Similarly, if the vertex x_p gets colored $\{1, 2, \ldots, k-1, 2k+1\}$, then x_p is considered false. We will now see that the assignment thus obtained for each x_p constitutes a satisfying assignment for ψ.

Assume for the sake of contradiction that the assignment is not a satisfying assignment for ψ. This means that there is a clause $C_j = x_{j,1} \vee x_{j,2} \vee x_{j,3}$ that is not satisfied. Here each literal $x_{j,1}, x_{j,2}$ and $x_{j,3}$ correspond to some x_p or $\overline{x_p}$. Since C_j is not satisfied, all of $x_{j,1}, x_{j,2}$ and $x_{j,3}$ are false literals. This means that $\overline{x_{j,1}}, \overline{x_{j,2}}$ and $\overline{x_{j,3}}$ are all true literals and are all colored with the set $\{1, 2, \ldots, k-1, 2k\}$.

Now let us consider the vertices $z_{j,1}, z_{j,2}$ and $z_{j,3}$. Because each of these vertices is adjacent to w and the set U', the free colors available for $z_{j,1}, z_{j,2}$ and $z_{j,3}$ are $\{k, k+1, \ldots, 2k\}$. Since $z_{j,1}, z_{j,2}$ and $z_{j,3}$ are adjacent to true literals, the color $2k$ is also ruled out, fixing the colors of $z_{j,1}, z_{j,2}$ and $z_{j,3}$ to the set $\{k, k+1, \ldots, 2k-1\}$.

Now we focus on the vertices $t_{j,1}, t_{j,2}$ and $t_{j,3}$. Because of its adjacencies to the set W', the free colors available to $t_{j,1}, t_{j,2}$ and $t_{j,3}$ are $[k-1] \cup \{2k-1, 2k, 2k+1\}$. Because each $t_{j,\ell}$ is adjacent to the corresponding $z_{j,\ell}$, the color $2k-1$ is ruled out, leaving only the colors $[k-1] \cup \{2k, 2k+1\}$ for $t_{j,1}, t_{j,2}$ and $t_{j,3}$. Recall that $t_{j,1}, t_{j,2}$ and $t_{j,3}$ form a totally independent 3-set of the Kneser graph Γ_j. Because of Theorem 18, the vertices $t_{j,1}, t_{j,2}$ and $t_{j,3}$ together require $k+2$ colors, since all of them share $k-1$ colors, with each getting a distinct k-th color. However, there are only $k+1$ free colors for $t_{j,1}, t_{j,2}$ and $t_{j,3}$, namely the set $[k-1] \cup \{2k, 2k+1\}$. This is a contradiction to the assumption that there is a clause that is not satisfied. Hence the assignment is a satisfying assignment for ψ.

Satisfiability implies colorability: To prove the reverse direction, let us assume that ψ is satisfiable. We will construct a $(2k+i, k, 0)$-coloring of G. We start with a satisfying assignment of ψ. For each $1 \leq p \leq n$, we color the vertices of the Kneser graph $A \cup B_p$ with their natural coloring. If the variable x_p is true in the satisfying assignment, we retain this coloring. If for some p, x_p is assigned false, then for that B_p, we swap the colors $2k$ and $2k+1$ for all the vertices. Notice that this is still a valid coloring of the Kneser graph $A \cup B_p$,

because the vertices in A will be unaffected by such a swap. For each variable x_p, this coloring assigns the vertex x_p with $\{1, 2, \ldots, k-1, 2k\}$ and the vertex $\overline{x_p}$ with $\{1, 2, \ldots, k-1, 2k+1\}$ if x_p is assigned true in the satisfying assignment, and vice versa if x_p is assigned false.

We assign vertex u the colors $\{k+1, k+2, \ldots, 2k\}$ and w the colors $\{1, 2, \ldots, k-1, 2k+1\}$. Now let us consider the vertices $z_{j,1}, z_{j,2}$ and $z_{j,3}$. If any of the $z_{j,\ell}$ is adjacent to a true literal x_p or $\overline{x_p}$, the vertex $z_{j,\ell}$ is assigned $\{k, k+1, \ldots, 2k-1\}$, since they are the only free colors available. However, if any of the $z_{j,\ell}$ is adjacent to a false literal, then colors from $\{k, k+1, \ldots, 2k\}$ are available.

Since we started with a satisfying assignment, every clause C_j contains at least one true literal, without loss of generality say $x_{j,1}$. The vertex $z_{j,1}$ is adjacent to the vertex $\overline{x_{j,1}}$ which is false. Now $z_{j,1}$ can be assigned the set $\{k, k+1, \ldots, 2k-2, 2k\}$ while $z_{j,2}$ and $z_{j,3}$ can be assigned the set $\{k, k+1, \ldots, 2k-1\}$.

The vertex $t_{j,1}$ has the colors $[k-1] \cup \{2k-1, 2k+1\}$ available while the vertices $t_{j,2}$ and $t_{j,3}$ have the colors $[k-1] \cup \{2k, 2k+1\}$ available for coloring. We can assign the sets $\{1, 2, \ldots, k-1, 2k-1\}$, $\{1, 2, \ldots, k-1, 2k\}$ and $\{1, 2, \ldots, k-1, 2k+1\}$ to the vertices $t_{j,1}, t_{j,2}$ and $t_{j,3}$ respectively. This is consistent with the coloring of a totally independent 3-set and hence can be extended to a $(k, 0)$-coloring of Γ_j.

Now all the vertices have been assigned colors, giving a $(k, 0)$-coloring of G.

We have completed the proof of correctness of the reduction. We have already seen that the size of G is polynomial in the size of ψ. Thus we have shown that the $(2k+i, k, 0)$-coloring problem is NP-complete for all $k, i \geq 1$. □

Acknowledgment: This work is partially supported by ECR/2017/001126 grant from SERB, DST, India. The authors would like to thank the anonymous reviewer for helpful comments, and pointing out a flaw in the proof of Theorem 12 in an earlier version of the paper.

References

[1] S. Joshi, S. Kalyanasundaram, A. S. Kare, S. Bhyravarapu, On the tractability of (k, i)-coloring, in: B. Panda, P. P. Goswami (Eds.), Algorithms and Discrete Applied Mathematics: Fourth International Conference, CALDAM 2018, India, 2018, pp. 188–198.

[2] I. Méndez-Díaz, P. Zabala, A generalization of the graph coloring problem, Investigation Operation 8 (1999) 167–184.

[3] S. Stahl, n-tuple colorings and associated graphs, Journal of Combinatorial Theory, Series B 20 (2) (1976) 185 – 203.

[4] B. Bollobás, A. Thomason, Set colourings of graphs, Discrete Mathematics 25 (1) (1979) 21 – 26.

[5] W. Klostermeyer, C. Q. Zhang, n-tuple coloring of planar graphs with large odd girth, Graphs and Combinatorics 18 (1) (2002) 119–132.

[6] P. Šparl, J. Žerovnik, A note on n-tuple colourings and circular colourings of planar graphs with large odd girth, International Journal of Computer Mathematics 84 (12) (2007) 1743–1746.

[7] R. W. Irving, NP-completeness of a family of graph-colouring problems, Discrete Applied Mathematics 5 (1) (1983) 111 – 117.

[8] D. Marx, Graph colouring problems and their applications in scheduling, Periodica Polytechnica Electrical Engineering 48 (1-2) (2004) 11–16.

[9] C. Beideman, J. Blocki, Set families with low pairwise intersection, arXiv preprint arXiv:1404.4622.

[10] R. C. Brigham, R. D. Dutton, Generalized k-tuple colorings of cycles and other graphs, Journal of Combinatorial Theory, Series B 32 (1) (1982) 90–94.

[11] F. Bonomo, G. Durán, I. Koch, M. Valencia-Pabon, On the (k, i)-coloring of cacti and complete graphs, Ars Combinatoria 137 (2018) 317 – 333.

[12] S. Johnson, A new upper bound for error-correcting codes, IRE Transactions on Information Theory 8 (3) (1962) 203–207.

[13] D. Majumdar, R. Neogi, V. Raman, P. Tale, Exact and parameterized algorithms for (k, i)-coloring, in: Algorithms and Discrete Applied Mathematics: Third International Conference, CALDAM 2017, India, 2017, pp. 281–293.

[14] V. Bafna, P. Berman, T. Fujito, A 2-approximation algorithm for the undirected feedback vertex set problem, SIAM Journal on Discrete Mathematics 12 (3) (1999) 289–297.

[15] Y. L. Wu, P. Austrin, T. Pitassi, D. Liu, Inapproximability of treewidth, one-shot pebbling, and related layout problems, J. Artif. Int. Res. 49 (1) (2014) 569–600.

[16] T. Kociumaka, M. Pilipczuk, Faster deterministic feedback vertex set, Information Processing Letters 114 (10) (2014) 556 – 560.

[17] S. Kratsch, P. Schweitzer, Isomorphism for graphs of bounded feedback vertex set number, Algorithm Theory-SWAT 2010 (2010) 81–92.

[18] B. M. Jansen, V. Raman, M. Vatshelle, Parameter ecology for feedback vertex set, Tsinghua Science and Technology 19 (4) (2014) 387–409.

[19] F. V. Fomin, D. Kratsch, Exact Exponential Algorithms, Texts in Theoretical Computer Science, An EATCS Series. Springer, 2010.

[20] J. W. Moon, L. Moser, On cliques in graphs, Israel Journal of Mathematics 3 (1) (1965) 23–28.

[21] M. Koivisto, An $O^*(2^n)$ algorithm for graph coloring and other partitioning problems via inclusion–exclusion, in: Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science, FOCS '06, IEEE Computer Society, Washington, DC, USA, 2006, pp. 583–590.

[22] P. Erdős, C. Ko, R. Rado, Intersection theorems for systems of finite sets, The Quarterly Journal of Mathematics 12 (1) (1961) 313–320.

[23] A. J. W. Hilton, E. C. Milner, Some intersection theorems for systems of finite sets, The Quarterly Journal of Mathematics 18 (1) (1967) 369–384.