On the Fine-Grained Complexity of Rainbow Coloring*

Łukasz Kowalik[1], Juho Lauri[2], and Arkadiusz Socała[3]

1 **University of Warsaw, Warsaw, Poland**
kowalik@mimuw.edu.pl
2 **Tampere University of Technology, Tampere, Finland**
juho.lauri@tut.fi
3 **University of Warsaw, Warsaw, Poland**
a.socala@mimuw.edu.pl

Abstract

The Rainbow k-Coloring problem asks whether the edges of a given graph can be colored in k colors so that every pair of vertices is connected by a rainbow path, i.e., a path with all edges of different colors. Our main result states that for any $k \geq 2$, there is no algorithm for Rainbow k-Coloring running in time $2^{o(n^{3/2})}$, unless ETH fails. Motivated by this negative result we consider two parameterized variants of the problem. In the Subset Rainbow k-Coloring problem, introduced by Chakraborty *et al.* [STACS 2009, J. Comb. Opt. 2009], we are additionally given a set S of pairs of vertices and we ask if there is a coloring in which all the pairs in S are connected by rainbow paths. We show that Subset Rainbow k-Coloring is FPT when parameterized by $|S|$. We also study Maximum Rainbow k-Coloring problem, where we are additionally given an integer q and we ask if there is a coloring in which at least q anti-edges are connected by rainbow paths. We show that the problem is FPT when parameterized by q and has a kernel of size $O(q)$ for every $k \geq 2$, extending the result of Ananth *et al.* [FSTTCS 2011]. We believe that our techniques used for the lower bounds may shed some light on the complexity of the classical Edge Coloring problem, where it is a major open question if a $2^{O(n)}$-time algorithm exists.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases graph coloring, computational complexity, lower bounds, exponential time hypothesis, FPT algorithms

Digital Object Identifier 10.4230/LIPIcs.ESA.2016.58

1 Introduction

The Rainbow k-Coloring problem asks whether the edges of a given graph can be colored in k colors so that every pair of vertices is connected by a rainbow path, i.e., a path with all edges of different colors. A minimum such k, called the *rainbow connection number* can be viewed as yet another measure of graph connectivity. The concept of rainbow coloring was introduced by Chartrand, Johns, McKeon, and Zhang [7] in 2008, while also featured in an earlier book of Chartrand and Zhang [8]. Chakraborty, Fischer, Matsliah, and Yuster [3] describe an interesting application of rainbow coloring in telecommunications. The problem

* Work partially supported by the National Science Centre of Poland, grant number 2013/09/B/ST6/03136 (Ł.K., A.S.), and by the Emil Aaltonen Foundation (J.L.). This is an extended abstract; for the full version see the technical report at arxiv [17].

is intensively studied from the combinatorial perspective, with over 100 papers published by now (see the survey of Li, Shi, and Sun [20] for an overview). However, the computational complexity of the problem seems less explored. It was conjectured by Caro, Lev, Roditty, Tuza, and Yuster [2] that the Rainbow k-Coloring problem is NP-complete for $k = 2$. This conjecture was confirmed by Chakraborty *et al.* [3]. Ananth, Nasre, and Sarpatwar [1] noticed that the proof of Chakraborty *et al.* in fact proves NP-completeness for every even $k > 1$, and complemented this by showing NP-completeness of the odd cases as well. An alternative hardness proof for every $k > 1$ was provided by Le and Tuza [19]. For complexity results on restricted graph classes, see e.g., [4, 5, 6, 12].

For many NP-complete graph problems there are algorithms running in time $2^{O(n)}$ for an n-vertex graph. This is obviously the case for problems asking for a set of vertices, like Clique or Vertex Cover, or more generally, for problems which admit polynomially (or even subexponentially) checkable $O(n)$-bit certificates. However, there are $2^{O(n)}$-time algorithms also for some problems for which such certificates are not known, including e.g., Hamiltonicity [13] and Vertex Coloring [18]. Unfortunately it seems that the best known worst-case running time bound for Rainbow k-Coloring is $k^m 2^k n^{O(1)}$, where m is the number of edges, which is obtained by checking each of the k^m colorings by a simple $2^k n^{O(1)}$-time dynamic programming algorithm [23]. Even in the simplest variant of just two colors, i.e., $k = 2$, this algorithm takes $2^{O(n^2)}$ time if the input graph is dense. It raises a natural question: is this problem really much harder than, say, Hamiltonicity, or have we just not found the right approach yet? Questions of this kind have received considerable attention recently. For example, the existence of a $2^{O(n)}$-time algorithm for Edge Coloring is a notorious question, appearing in numerous open problem lists. On the other hand, it was shown that unless the Exponential Time Hypothesis fails, there is no algorithm running in time $2^{o(n \log n)}$ for Channel Assignment [21], Subgraph Homomorphism, and Subgraph Isomorphism [9]. Let us recall the precise statement of the Exponential Time Hypothesis (ETH).

▶ **Hypothesis 1** (Exponential Time Hypothesis [14]). *There exists a constant $c > 0$, such that there is no deterministic algorithm solving* 3-SAT *in time $O^*(2^{cn})$.*

Note that some kind of a complexity assumption, like ETH, is hard to avoid when we prove exponential lower bounds, unless one aims at proving P $\neq$ NP.

Main Result. Our main result is the following theorem.

▶ **Theorem 2.** *For any $k \geq 2$,* Rainbow k-Coloring *can be solved neither in $2^{o(n^{3/2})}$ nor $2^{o(m/\log m)}$ time where n and m are the number of vertices and edges respectively, unless ETH fails.*

Hence, this is an NP-complete graph problem which does not admit a $2^{o(n^{1+\epsilon})}$-time algorithm (under reasonable complexity assumptions), for an $\epsilon > 0$. Such lower bounds are fairly rare in the literature. The best known algorithm for Rainbow k-Coloring just verifies all possible colorings and thus it runs in time $2^{O(m)}$ for any fixed k. Our lower bounds mean that one cannot hope for substantial improvements in this running time.

Remaining Lower Bounds. We also study a natural generalized problem, called Subset Rainbow k-Coloring, introduced by Chakraborty *et al.* [3] as a natural intermediate step in reductions from 3-SAT to Rainbow k-Coloring. In Subset Rainbow k-Coloring, we are given a connected graph G, and a set of pairs of vertices $S \subseteq \binom{V(G)}{2}$. Elements of S

are called *requests*. For a given coloring of $E(G)$ we say that a request $\{u, v\}$ is *satisfied* if u and v are connected by a rainbow path. The goal in Subset Rainbow k-Coloring is to determine whether there is a k-coloring of $E(G)$ such that every pair in S is satisfied. Our main result implies that Subset Rainbow k-Coloring admits no algorithm running in time $2^{o(n^{3/2})}$, under ETH. We show also two more lower bounds, as follows.

▶ **Theorem 3.** *For any $k \geq 2$,* Subset Rainbow k-Coloring *can be solved neither in time $2^{o(n^{3/2})}$, nor in time $2^{o(m)}$, nor in time $2^{o(s)}$ where n is the number of vertices, m is the number of edges, and s is the number of requests, unless ETH fails.*

An interesting feature here is that for $k = 2$ the $2^{o(m)}$ and $2^{o(s)}$ bounds are *tight* up to a polynomial factor (a $2^m n^{O(1)}$ algorithm is immediate, and a $2^{|S|}n^{O(1)}$-time algorithm is discussed in the next paragraph).

New Algorithms. In the context of the hardness results mentioned above it is natural to ask for FPT algorithms for Subset Rainbow k-Coloring. We show that for every fixed k, Subset Rainbow k-Coloring parameterized by $|S|$ is FPT:

▶ **Theorem 4.** *For every integer k,* Subset Rainbow k-Coloring *is FPT and it has an algorithm running in time $|S|^{O(|S|)}n^{O(1)}$.*

For the 2 color case we are able to show a different, faster algorithm running in time $2^{|S|}n^{O(1)}$, which is tight up to a polynomial factor.

We also study the Maximum Rainbow k-Coloring problem, introduced by Ananth, Nasre, and Sarpatwar [1]. Intuitively, the idea is to parameterize the problem by the number of pairs to satisfy. However, all pairs of adjacent vertices are trivially satisfied by any edge-coloring. Hence, we parameterize by the number of anti-edges to satisfy. More formally, in Maximum Rainbow k-Coloring we are given a graph $G = (V, E)$, an integer q, and asked whether there is a coloring of E that satisfies at least q anti-edges. First, we show that the maximization version of the problem (find maximum such q) admits a constant factor approximation algorithm for every fixed value of k. Second, we show that Maximum Rainbow k-Coloring is FPT for every $k \geq 2$, which generalizes the result of Ananth *et al.* [1] who showed this claim for the $k = 2$ case. Our algorithm runs in time $2^{q \log q}n^{O(1)}$ for any k, which is faster than the algorithm of Ananth *et al.* for 2 colors. For 2 colors we give an even faster algorithm, running in time $8^q n^{O(1)}$. We also show that the problem admits a kernel size $O(q)$, i.e., that there is a polynomial-time algorithm that returns an equivalent instance with $O(q)$ vertices. (For more background on kernelization see e.g., [10].) Before, this was known only for $k = 2$ (due to Ananth *et al.* [1]). Our main results for Maximum Rainbow k-Coloring are summarized in the following theorem.

▶ **Theorem 5.** Maximum Rainbow k-Coloring *parameterized by the number of anti-edges q is FPT for every $k \geq 2$. Moreover, it admits a kernel of linear size.*

Notation. For standard graph-theoretic notions, we refer the reader to [11]. All graphs we consider in this paper are simple and undirected. We denote $\Delta_1(G) = \max\{\Delta(G), 1\}$.

By $\bar{E}$ we denote the set of anti-edges, i.e., $\bar{E} = \binom{V}{2} \setminus E$. When $G = (V, E)$ is a graph then $\bar{G} = (V, \bar{E})$ is its *complement graph*. By $x^{\underline{k}}$ we denote the falling factorial, i.e., $x^{\underline{k}} = x(x-1)\cdots(x-k+1)$. For an integer k, we denote $[k] = \{1, \ldots, k\}$. For a (partial) function c, by $\mathrm{Dom}(c)$ we denote its domain.

If I and J are instances of decision problems P and R, respectively, then we say that I and J are *equivalent*, when either both I and J are YES-instances or both are NO-instances.

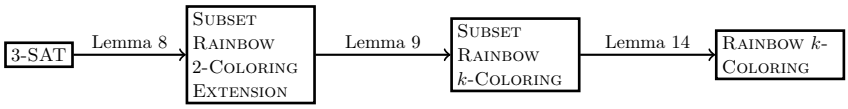

Figure 1 A simplified road map of our reductions.

Organization of the paper. In Section 2 we present our hardness results. The main difficulties we encountered are sketched at the beginning of that section. Due to space constraints proofs of the claims marked by ★ are skipped and can be found in the full version [17]. Next, in Section 3 we present our algorithms for Subset Rainbow k-Coloring. Again because of the space limitations, our algorithms for Maximum Rainbow k-Coloring are skipped in this extended abstract and are available in the full version [17].

2 Hardness of rainbow coloring

2.1 Overview

The main goal of this section is to show that for any $k \geq 2$ Rainbow k-Coloring does not admit an algorithm running in time $2^{o(n^{3/2})}$, unless the Exponential Time Hypothesis fails. Let us give a high-level overview of our proof. A natural idea would be to begin with a 3-SAT formula ϕ with n variables and then transform it in time $2^{o(n)}$ to an equivalent instance $G = (V, E)$ of Rainbow k-Coloring with $O(n^{2/3})$ vertices. Then indeed a $2^{o(|V|^{3/2})}$-time algorithm that solves Rainbow 2-Coloring can be used to decide 3-SAT in time $2^{o(n)}$. Note that in a typical NP-hardness reduction, we observe some polynomial blow-up of the instance size. For example, one can verify that in the reduction of Chakraborty *et al.* [3], the initial 3-SAT formula with n variables and m clauses is transformed into a graph with $\Theta(n^4 + m^4)$ vertices and edges. In our case, instead of a blow-up we aim at *compression*: the number of vertices needs to be much smaller than the number of variables in the input formula ϕ. As usual in reductions, variables and clauses in ϕ are going to correspond to some structures in G, called gadgets. The compression requirement means that our gadgets need to share vertices. To make our lives slightly easier, we apply the following well-known Sparsification Lemma, which allows for assuming that the number of clauses is $O(n)$.

▶ **Lemma 6** (Sparsification Lemma [15]). *For each $\varepsilon > 0$ there exist a constants c_ε, such that any* 3-SAT *formula φ with n variables can be expressed as $\varphi = \vee_{i=1}^{t} \psi_i$, where $t \leq 2^{\varepsilon n}$ and each ψ_i is a* 3-SAT *formula with the same variable set as φ, but contains at most $c_\varepsilon n$ clauses. Moreover, this disjunction can be computed in time $O^*(2^{\varepsilon n})$.*

Note that by using the Sparsification Lemma we tweak our general plan a bit: instead of creating one equivalent instance, we are going to create $2^{\varepsilon n}$ instances (for arbitrarily small ϵ), each with $O(n^{2/3})$ vertices. The following lemma further simplifies the instance.

▶ **Lemma 7** ([22]). *Given a* 3-SAT *formula φ with m clauses one can transform it in polynomial time into a formula φ' with $O(m)$ variables and $O(m)$ clauses, such that φ' is satisfiable iff φ' is satisfiable, and moreover each clause of φ' contains exactly three different variables and each variable occurs in at most* 4 *clauses of φ'.*

Now our goal is to transform a 3-SAT formula ϕ with n variables such that every variable occurs in at most 4 clauses, to a graph with $O(n^{2/3})$ vertices — an equivalent instance of Rainbow k-Coloring. We do it in three steps (see Fig 1).

In the first step we transform ϕ to an instance $I = (G, S, c_0)$ of Subset Rainbow 2-Coloring Extension, which is a generalization of Subset Rainbow 2-Coloring, where c_0, called a *precoloring*, is a partial coloring of the edges of G into two colors and the goal is to determine if there is an edge-coloring of $E(G)$ which extends c_0 and such that all pairs of S are satisfied. The first step is crucial, because here the compression takes place: $|V(G)| = O(n^{2/3})$ and $E(G) = O(n)$. The major challenge in the construction is avoiding interference between gadgets that share a vertex: to this end we define various conflict graphs and we show that they can be vertex-colored in a few colors. This reduction is described in Section 2.2.

In the second step (Lemma 9) we reduce Subset Rainbow 2-Coloring Extension to Subset Rainbow k-Coloring, for every $k \geq 2$. In fact, in the full version this is done in two sub-steps, via Subset Rainbow k-Coloring Extension. The number of the vertices in the resulting instance does not increase more than by a constant factor. This step is rather standard, though some technicalities appear because we need to guarantee additional properties of the output instance, which are needed by the reduction in the third step.

The last step (Section 2.3), where we reduce an instance $(G = (V, E), S)$ of Subset Rainbow k-Coloring to an instance G' of Rainbow k-Coloring, is yet another challenge. We would like to get rid of the set of requests somehow. For simplicity, let us focus on the $k = 2$ case now. Here, the natural idea, used actually by Chakraborty *et al.* [3] is to create, for every $\{u, v\} \notin S$, a path (u, x_{uv}, v) through a new vertex x_{uv}. Such a path cannot help any of the requests $\{u', v'\} \in S$ to get satisfied (since if it creates a new path P' between u' and v', then P' has length at least 3), and by coloring it into two different colors we can satisfy $\{u, v\}$. Unfortunately, in our case we cannot afford for creating a new vertex for every such $\{u, v\}$, because that would result in a quadratic blow up in the number of vertices. However, one can observe that for any biclique (a complete bipartite subgraph) in the graph $(V, \binom{V}{2} \setminus S)$ it is sufficient to use just one such vertex x (connected to all the vertices of the biclique). By applying a result of Jukna [16] we can show that in our specific instance of Subset Rainbow 2-Coloring which results from a 3-SAT formula, the number of bicliques needed to cover all the pairs in $\binom{V}{2} \setminus S$ is small enough. We show a $2^{|V(G)|}|V(G)|^{O(1)}$-time algorithm to find such a cover. Although this algorithm does not seem fast, in our case $|V(G)| = O(n^{2/3})$, so this complexity is *subexponential* in the number of variables of the input formula, which is enough for our goal. The case of $k \geq 3$ is similar, i.e., we also use the biclique cover. However, the details are much more technical because for each biclique we need to introduce a much more complex gadget.

2.2 From 3-SAT to Subset Rainbow k-Coloring

Let Subset Rainbow k-Coloring Extension be a generalization of Subset Rainbow k-Coloring, where c_0 is a partial k-coloring of the edges of G and the goal is to determine if there is an edge-coloring of $E(G)$ which extends c_0 and such that all pairs of S are satisfied. In this section we show a reduction (Lemma 8) from 3-SAT to Subset Rainbow 2-Coloring Extension.

For an instance $I = (G, S, c_0)$ of Subset Rainbow k-Coloring Extension (for any $k \geq 2$), let us define a *precoloring conflict graph* CG_I. Its vertex set is the set of colored edges, i.e., $V(CG_I) = \mathrm{Dom}(c_0)$. Two different colored edges e_1 and e_2 (treated as vertices of CG_I) are adjacent in CG_I when they are incident in G or there is a pair of endpoints $u \in e_1$ and $v \in e_2$ such that $uv \in E(G) \cup S$.

In what follows the reduction in Lemma 8 is going to be pipelined with further reductions going through Subset Rainbow k-Coloring Extension and Subset Rainbow k-

Coloring to Rainbow k-Coloring. In these three reductions we need to keep the instance small. To this end, the instance of Subset Rainbow 2-Coloring Extension resulting in Lemma 8 has to satisfy some additional properties, which are formulated in the claim of Lemma 8. Their role will become clearer later on.

▶ **Lemma 8.** *Given a* 3-SAT *formula φ with n variables such that each clause of φ contains exactly three variables and each variable occurs in at most four clauses, one can construct in polynomial time an equivalent instance (G, S, c_0) of* Subset Rainbow 2-Coloring Extension *such that G has $O(n^{2/3})$ vertices and $O(n)$ edges. Moreover, $\Delta(G) = O(n^{1/3})$, $\Delta(V(G), S) = O(n^{1/3})$, $|\mathrm{Dom}(c_0)| = O(n^{2/3})$ and along with the instance $I = (G, S, c_0)$ the algorithm constructs a proper vertex 4-coloring of $(V(G), E \cup S)$ (so also of $(V(G), S)$) and a proper vertex $O(n^{1/3})$-coloring of the precoloring conflict graph CG_I.*

Proof. Let m denote the number of clauses in φ. Observe that $m \leq \frac{4}{3}n$. Let Var and Cl denote the sets of variables and clauses of φ. For more clarity, the two colors of the partial coloring c_0 will be called T and F. Let us describe the graph G along with a set of anti-edges S. Graph G consists of two disjoint vertex subsets: the variable part and the clause part. The intuition is that in any 2-edge coloring of G that extends c_0 and satisfies all pairs in S

- edge colors in the variable part represent an assignment of the variables of φ,
- edge colors in the clause part represent a choice of literals that satisfy all the clauses, and
- edge colors between the two parts make the values of the literals from the clause part consistent with the assignment represented by the variable part.

The variable part. The vertices of the variable part consist of the *middle set* M and $\lceil n^{1/3} \rceil$ layers $L_1 \cup L_2 \cdots \cup L_{\lceil n^{1/3} \rceil}$. The middle set M consists of vertices m_i for each $i = 1, \ldots, \lceil n^{2/3} \rceil + 9$. For every $i = 1, \ldots, \lceil n^{1/3} \rceil$ the layer L_i consists of two parts: upper $L_i^{\uparrow} = \{u_{i,j} \ : \ j = 1, \ldots, \lceil n^{1/3} \rceil + 3\}$ and lower $L_i^{\downarrow} = \{l_{i,j} \ : \ j = 1, \ldots, \lceil n^{1/3} \rceil + 3\}$.

We are going to define four functions: $\mathrm{mid} : \mathrm{Var} \to M$, $\mathrm{lay}, \mathrm{up}, \mathrm{low} : \mathrm{Var} \to [\lceil n^{1/3} \rceil]$. Then, for every variable $x \in \mathrm{Var}$ we add two edges $u_{\mathrm{lay}(x),\mathrm{up}(x)}\mathrm{mid}(x)$ and $\mathrm{mid}(x)l_{\mathrm{lay}(x),\mathrm{low}(x)}$. Moreover, we add the pair $p_x = \{u_{\mathrm{lay}(x),\mathrm{up}(x)}, l_{\mathrm{lay}(x),\mathrm{low}(x)}\}$ to S. In other words, x corresponds to the 2-path $u_{\mathrm{lay}(x),\mathrm{up}(x)}\mathrm{mid}(x)l_{\mathrm{lay}(x),\mathrm{low}(x)}$. Now we describe a careful construction of the four functions, that guarantee several useful properties (for example edge-disjointness of paths corresponding to different variables).

Let us define the *variable conflict graph* $G_V = (\mathrm{Var}, E_{G_V})$, where for two variables $x, y \in \mathrm{Var}$ we have xy are adjacent iff they both occur in the same clause. Since every variable occurs in at most 4 clauses, $\Delta(G_V) \leq 8$. It follows that there is a proper vertex 9-coloring $\alpha : Var \to [9]$ of G_v, and it can be found by a simple linear time algorithm. Next, each of the 9 color classes $\alpha^{-1}(i)$ is partitioned into $\left\lceil |\alpha^{-1}(i)| / \lceil n^{1/3} \rceil \right\rceil$ disjoint groups, each of size at most $\lceil n^{1/3} \rceil$. It follows that the total number n_g of groups is at most $\lceil n^{2/3} \rceil + 9$. Let us number the groups arbitrarily from 1 to n_g and for every variable $x \in \mathrm{Var}$, let $g(x)$ be the number of the group that contains x. Then we define $\mathrm{mid}(x) = m_{g(x)}$. Since any group contains only vertices of the same color we can state the following property:
($\mathbf{P_1}$) If variables x and y occur in the same clause then $\mathrm{mid}(x) \neq \mathrm{mid}(y)$.

Now, for every variable x we define its layer, i.e., the value of the function $\mathrm{lay}(x)$. Recall that for every $i = 1, \ldots, \lceil n^{2/3} \rceil + 9$ the i-th group $\mathrm{mid}^{-1}(m_i)$ contains at most $\lceil n^{1/3} \rceil$ variables. Inside each group, number the variables arbitrarily and let $\mathrm{lay}(x)$ be the number of variable x in its group, $\mathrm{lay}(x) \in [n^{1/3}]$. This implies another important property.
($\mathbf{P_2}$) If variables x and y belong to the same layer then $\mathrm{mid}(x) \neq \mathrm{mid}(y)$.

Observe that every layer gets assigned at most $\lceil n^{2/3} \rceil + 9$ variables. For every layer L_i pick any injective function $h_i : \text{lay}^{-1}(i) \to [\lceil n^{1/3} \rceil + 3]^2$. Then, for every variable $x \in \text{Var}$ we put $(\text{up}(x), \text{low}(x)) = h_{\text{lay}(x)}(x)$. Note that by (P_2) we have the following.

(P_3) For every variable x there is exactly one 2-path in G connecting p_x, namely $(u_{\text{lay}(x),\text{up}(x)}, \text{mid}(x), l_{\text{lay}(x),\text{low}(x)})$.

(P_4) For every pair of variables x, y the two unique paths connecting p_x and p_y are edge-disjoint.

Although we are going to add more edges and vertices to G, none of these edges has any endpoint in $\bigcup_i L_i$, so P_3 will stay satisfied.

The clause part. The vertices of the clause part are partitioned into $O(m^{1/3})$ clusters. Similarly as in the case of variables, each clause is going to correspond to a pair of vertices in the same cluster. Again, the assignment of clauses to clusters has to be done carefully. To this end we introduce the *clause conflict graph* $G_C = (\text{Cl}, E_{G_C})$. Two different clauses C_1 and C_2 are adjacent in G_C if C_1 contains a variable x_1 and C_2 contains a variable x_2 such that $\text{mid}(x_1) = \text{mid}(x_2)$. Fix a variable x_1. Since $|\text{mid}^{-1}(\text{mid}(x_1))| \leq \lceil n^{1/3} \rceil$, there are at most $\lceil n^{1/3} \rceil$ variables x_2 such that $\text{mid}(x_1) = \text{mid}(x_2)$. Since every clause contains 3 variables, and each of them is in at most 4 clauses, $\Delta(G_C) \leq 12 \lceil n^{1/3} \rceil$. It follows that in polynomial time we can find a proper coloring β of the vertices of G_C into at most $12 \lceil n^{1/3} \rceil + 1$ colors. Moreover, if for any color j its color class $\beta^{-1}(j)$ is larger than $\lceil n^{2/3} \rceil$ we partition it into $\lceil |\beta^{-1}(j)| / \lceil n^{2/3} \rceil \rceil$ new colors. Clearly, in total we produce at most $\frac{4}{3} \lceil n^{1/3} \rceil$ new colors in this way because $m \leq \frac{4}{3} n \leq \frac{4}{3} \lceil n^{1/3} \rceil \cdot \lceil n^{2/3} \rceil$. Hence, in what follows we assume that each color class of β is of size at most $\lceil n^{2/3} \rceil$, and the total number of colors $s \leq 14 \lceil n^{1/3} \rceil + 1$. In what follows we construct s clusters $Q_1, \ldots, Q_s$. Every clause $C \in \text{Cl}$ is going to correspond to a pair of vertices in the cluster $Q_{\beta(C)}$.

Fix $i = 1, \ldots, s$. Let us describe the subgraph induced by cluster Q_i. Define *cluster conflict graph* $G_i = (\beta^{-1}(i), E_{G_i})$. Two different clauses $C_1, C_2 \in \beta^{-1}(i)$ are adjacent in G_i if there are three variables x_1, x_2, and x_3 such that (i) C_1 contains x_1, (ii) C_2 contains x_2, (iii) $(\text{lay}(x_1), \text{up}(x_1)) = (\text{lay}(x_3), \text{up}(x_3))$ and (iv) $\text{mid}(x_2) = \text{mid}(x_3)$. Fix a variable x_1 which appears in a clause $C_1 \in \beta^{-1}(i)$. By our construction, there are at most $\lceil n^{1/3} \rceil + 2$ other variables x_3 that map to the same pair as x_1 by functions lay and up. For each such x_3 there are at most $\lceil n^{1/3} \rceil$ variables x_2 such that $\text{mid}(x_2) = \text{mid}(x_3)$; however, at most one of these variables belongs to a clause C_2 from the same cluster $\beta^{-1}(i)$, by the definition of the coloring β. It follows that $\Delta(G_i) \leq 12(\lceil n^{1/3} \rceil + 2)$. Hence in polynomial time we can find a proper coloring γ_i of the vertices of G_i into at most $12(\lceil n^{1/3} \rceil + 2) + 1$ colors. Similarly as in the case of the coloring β, we can assume that each of the color classes of γ_i has at most $\lceil n^{1/3} \rceil$ clauses, at the expense of at most $\lceil n^{1/3} \rceil$ additional colors. It follows that we can construct in polynomial time a function $g : \text{Cl} \to [\lceil n^{1/3} \rceil]$ such that for every cluster $i = 1, \ldots, s$ and for every color class S of γ_i g is injective on S. Let $n_i \leq 13 \lceil n^{1/3} \rceil + 25$ be the number of colors used by γ_i. For notational convenience, let us define a function $\gamma : \text{Cl} \to [\max_i n_i]$ such that for any clause C we have $\gamma(C) = \gamma_{\beta(C)}(C)$.

We are ready to define the vertices and edges of Q_i. It is a union of three disjoint vertex sets A_i, B_i, and C_i. We have $A_i = \{a_{i,j} \ : \ j = 1, \ldots, \lceil n^{1/3} \rceil\}$, $B_i = \{b^k_{i,j} \ : \ j = 1, \ldots, n_i, k = 1, 2, 3\}$, and $C_i = \{c_{i,j} \ : \ j = 1, \ldots, n_i\}$. For every $j = 1, \ldots, n_i$ and for every $k = 1, 2, 3$ we add edge $c_{i,j} b^k_{i,j}$ to G, and we color it by c_0 to color F. (These are the only edges pre-colored in the whole graph G.) For every clause $C \in \beta^{-1}(i)$ we do the following. For each $k = 1, 2, 3$, add the edge $(a_{i,g(C)}, b^k_{i,\gamma(C)})$ to G. Finally, add the pair $\{a_{i,g(C)}, c_{i,\gamma(C)}\}$ to S. Clearly, the following holds:

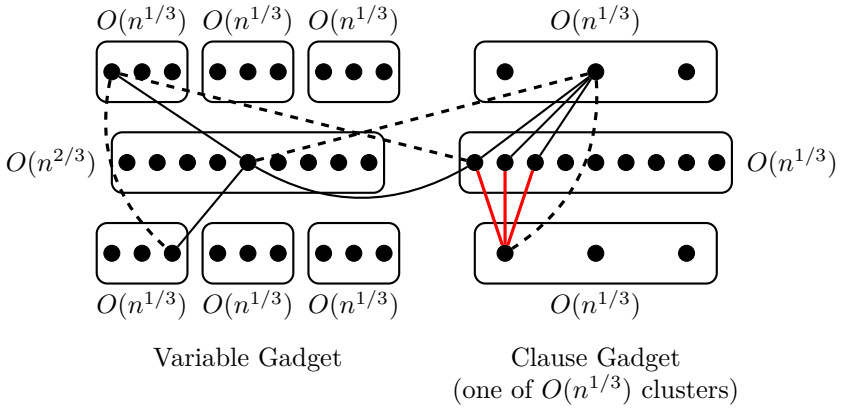

Figure 2 A simplified view of the obtained instance. Edges (solid lines) and requests (dashed lines) representing one variable and one clause that contains this variable are presented on the picture.

($\mathbf{P_5}$) Let C be any clause. Let $i = \beta(C)$ and let $j = g(C)$. Then there are exactly three 2-paths between $a_{\beta(C),g(C)}$ and $c_{\beta(C),\gamma(C)}$, each going through $b^k_{\beta(C),\gamma(C)}$ for $k = 1, 2, 3$.

Connections between the two parts. Consider a clause $C = \{\ell_1, \ell_2, \ell_3\}$ and its k-th literal ℓ_k for each $k = 1, 2, 3$. Then for some variable x we have $\ell_k = x$ or $\ell_k = \bar{x}$. We add the edge $b^k_{\beta(C),\gamma(C)}\mathrm{mid}(x)$ and we add the pair $\{\mathrm{mid}(x), a_{\beta(C),g(C)}\}$ to S. If $\ell_k = x$, we also add the pair $\{b^k_{\beta(C),\gamma(C)}, u_{\mathrm{lay}(x),\mathrm{up}(x)}\}$ to S; otherwise we add the pair $\{b^k_{\beta(C),\gamma(C)}, l_{\mathrm{lay}(x),\mathrm{low}(x)}\}$ to S. We claim the following.

($\mathbf{P_6}$) Every edge between the two parts was added exactly once, i.e., for every edge uv such that u is in the clause part and v is in the variable part, there is exactly one clause C and exactly one literal $\ell_k \in C$ such that $u = b^k_{\beta(C),\gamma(C)}$ and $v = \mathrm{mid}(x)$, where x is the variable in ℓ_k.

Indeed, assume for a contradiction that there is a clause C_1 with its k_1-th literal containing x_1 and a clause C_2 with its k_2-th literal containing x_2 such that $b^{k_1}_{\beta(C_1),\gamma(C_1)} = b^{k_2}_{\beta(C_2),\gamma(C_2)}$ and $\mathrm{mid}(x_1) = \mathrm{mid}(x_2)$. Then $C_1 \neq C_2$ by (P_1). Since $\mathrm{mid}(x_1) = \mathrm{mid}(x_2)$, C_1 and C_2 are adjacent in the clause conflict graph G_C. It follows that $\beta(C_1) \neq \beta(C_2)$, so two different clusters share a vertex, a contradiction.

This finishes the description of the instance (G, S, c_0). (See Fig. 2.)

From an assignment to a coloring. Let $\xi : \mathrm{Var} \to \{T, F\}$ be a satisfying assignment of φ. We claim that there is a coloring c of $E(G)$ which extends c_0 and satisfies all pairs in S. We define c as follows. Denote $\overline{F} = T$, $\overline{T} = F$ and $\xi(\bar{x}) = \overline{\xi(x)}$. For every variable $x \in \mathrm{Var}$ we put $c(u_{\mathrm{lay}(x),\mathrm{up}(x)}\mathrm{mid}(x)) = \xi(x)$ and $c(\mathrm{mid}(x)l_{\mathrm{lay}(x),\mathrm{low}(x)}) = \overline{\xi(x)}$. By ($\mathrm{P}_3$) and ($\mathrm{P}_4$) each edge is colored exactly once. Note that it satisfies all the pairs in S between vertices in the variable part.

For each clause C and each of its literals ℓ_k do the following. Let us color the edge $a_{\beta(C),g(C)}b^k_{\beta(C),\gamma(C)}$ with the color $\xi(\ell_k)$. Since g is injective on color classes of $\gamma_{\beta(C)}$, after processing all the literals in all the clauses, no edge is colored more than once. Recall that for every clause C we added exactly one pair to S, namely $\{a_{\beta(C),g(C)}, c_{\beta(C),\gamma(C)}\}$. Pick any of C's satisfied literals, say ℓ_k. Note that the pair $\{a_{\beta(C),g(C)}, c_{\beta(C),\gamma(C)}\}$ is then satisfied,

because edge $a_{\beta(C),g(C)}b^k_{\beta(C),\gamma(C)}$ is colored by T and $b^k_{\beta(C),\gamma(C)}c_{\beta(C),\gamma(C)}$ is colored by F. Hence all the pairs in S between vertices in the clause part are satisfied.

Now let us color the edges between the clause part and the variable part. Consider any such edge uv, i.e., u is in the clause part and v is in the variable part. By (P_6), there is exactly one clause C and exactly one literal $\ell_k \in C$ such that $u = b^k_{\beta(C),\gamma(C)}$ and $v = \mathrm{mid}(x)$, where x is the variable in ℓ_k. Color the edge $b^k_{\beta(C),\gamma(C)}\mathrm{mid}(x)$ with the color $\overline{\xi(\ell_k)}$. Then the pair $\{\mathrm{mid}(x), a_{\beta(C),g(C)}\}$ is satisfied by the path $(\mathrm{mid}(x), b^k_{\beta(C),\gamma(C)}, a_{\beta(C),g(C)})$, since $c(b^k_{\beta(C),\gamma(C)}a_{\beta(C),g(C)}) = \xi(\ell_k)$. Assume $\ell_k = x$. Then the pair $\{b^k_{\beta(C),\gamma(C)}, u_{\mathrm{lay}(x),\mathrm{up}(x)}\}$ is satisfied by the path $(b^k_{\beta(C),\gamma(C)}, \mathrm{mid}(x), u_{\mathrm{lay}(x),\mathrm{up}(x)})$, since its first edge is colored by $\overline{\xi(\ell_k)} = \overline{\xi(x)}$ and its second edge is colored by $\xi(x)$. Analogously, when $\ell_k = \bar{x}$, then the pair $\{b^k_{\beta(C),\gamma(C)}, l_{\mathrm{lay}(x),\mathrm{low}(x)}\}$ is satisfied by the path $(b^k_{\beta(C),\gamma(C)}, \mathrm{mid}(x), l_{\mathrm{lay}(x),\mathrm{low}(x)})$, since its first edge is colored by $\overline{\xi(\ell_k)} = \xi(x)$ and its second edge is colored by $\overline{\xi(x)}$.

It follows that we colored all the edges and all the pairs in S are satisfied, so (G, S, c_0) is a YES-instance, as required.

From a coloring to an assignment. Let $c : E(G) \to \{T, F\}$ be a coloring which extends c_0 and satisfies all pairs in S. Consider the following variable assignment: for every $x \in \mathrm{Var}$, we put $\xi(x) = c(u_{\mathrm{lay}(x),\mathrm{up}(x)}\mathrm{mid}(x))$. We claim that ξ satisfies all the clauses of φ. Consider an arbitrary clause $C = \{\ell_1, \ell_2, \ell_3\}$.

Since the pair $\{a_{\beta(C),g(C)}, c_{\beta(C),\gamma(C)}\}$ is satisfied, there is a 2-color 2-path P between $a_{\beta(C),g(C)}$ and $c_{\beta(C),\gamma(C)}$. Recall that $N(c_{\beta(C),\gamma(C)}) = \{b^k_{\beta(C),\gamma(C)} : k = 1, 2, 3\}$, so there is $k = 1, 2, 3$ such that $b^k_{\beta(C),\gamma(C)}$ is the internal vertex on P. Since c extends c_0 and $c_0(b^k_{\beta(C),\gamma(C)}c_{\beta(C),\gamma(C)}) = F$, we infer that $c(a_{\beta(C),g(C)}b^k_{\beta(C),\gamma(C)}) = T$. Let x be the variable in the literal ℓ_k.

Since the pair $\{\mathrm{mid}(x), a_{\beta(C),g(C)}\}$ is satisfied, there is a 2-color 2-path Q between $\mathrm{mid}(x)$ and $a_{\beta(C),g(C)}$. Then the internal vertex of Q is $b^{k'}_{\beta(C'),\gamma(C')}$, for some clause C' and integer $k' = 1, 2, 3$. Let y be the variable in the k'-th literal of C'. Since there is an edge between $\mathrm{mid}(x)$ and $b^{k'}_{\beta(C'),\gamma(C')}$, from (P_6) we infer that $\mathrm{mid}(y) = \mathrm{mid}(x)$. If $C = C'$ and $k' \neq k$, then by (P_1) we get that $\mathrm{mid}(x) \neq \mathrm{mid}(y)$, a contradiction. If $C \neq C'$, since $\mathrm{mid}(y) = \mathrm{mid}(x)$, the clauses C and C' are adjacent in the clause conflict graph G_C, so $\beta(C') \neq \beta(C)$. However, then the edge $b^{k'}_{\beta(C'),\gamma(C')}a_{\beta(C),g(C)}$ of Q goes between two clusters, a contradiction. Hence $C' = C$ and $k' = k$, i.e., $Q = (\mathrm{mid}(x), b^k_{\beta(C),\gamma(C)}, a_{\beta(C),g(C)})$. Since $c(b^k_{\beta(C),\gamma(C)}a_{\beta(C),g(C)}) = T$, we get $c(\mathrm{mid}(x)b^k_{\beta(C),\gamma(C)}) = F$. Now assume w.l.o.g. that $\ell_k = x$, the case $\ell_k = \bar{x}$ is analogous.

Since the pair $\{b^k_{\beta(C),\gamma(C)}, u_{\mathrm{lay}(x),\mathrm{up}(x)}\}$ is satisfied, there is a 2-color 2-path R between $b^k_{\beta(C),\gamma(C)}$ and $u_{\mathrm{lay}(x),\mathrm{up}(x)}$. Then the internal vertex z of R belongs to M. By (P_6) there is a literal ℓ_k which belongs to a clause C_2 and contains a variable x_2 such that $z = \mathrm{mid}(x_2)$ and $b^k_{\beta(C),\gamma(C)} = b^k_{\beta(C_2),\gamma(C_2)}$. In particular, $\beta(C) = \beta(C_2)$ and $\gamma(C) = \gamma(C_2)$. Assume $C_2 \neq C$. There is a variable, say x_3, corresponding to edge $\mathrm{mid}(x_2)u_{\mathrm{lay}(x),\mathrm{up}(x)}$, i.e., $\mathrm{mid}(x_2) = \mathrm{mid}(x_3)$ and $u_{\mathrm{lay}(x),\mathrm{up}(x)} = u_{\mathrm{lay}(x_3),\mathrm{up}(x_3)}$. It follows that C and C_2 are adjacent in $G_{\beta(C)}$, which contradicts the fact that $\gamma(C) = \gamma(C_2)$. Hence $C_2 = C$, i.e., there is exactly one 2-path between $b^k_{\beta(C),\gamma(C)}$ and $u_{\mathrm{lay}(x),\mathrm{up}(x)}$, and it goes through $\mathrm{mid}(x)$. Since $c(\mathrm{mid}(x)b^k_{\beta(C),\gamma(C)}) = F$ and the path is 2-color, we get that $c(u_{\mathrm{lay}(x),\mathrm{up}(x)}\mathrm{mid}(x)) = T$. Hence $\xi(\ell_k) = \xi(x) = T$, so clause C is satisfied, as required.

It finishes the proof. (The analysis of the size of the resulting instance and its other properties described in the claim is not immediate; because of the space constraints we skip them here.) ◀

The proof of the following lemma is non-trivial, but standard (see lemmas 4 and 5 in the full version [17].)

▶ **Lemma 9 (★).** *For any fixed $k \geq 2$, there is a polynomial time algorithm which given an instance $I = (G = (V, E), S, c_0)$ of* Subset Rainbow 2-Coloring Extension *constructs an equivalent instance $(G' = (V', E'), S')$ of* Subset Rainbow k-Coloring *such that $|V'| = O(k|V|k^2\ell)$, $|E'| = |E|+O(k|V|)+|\mathrm{Dom}(c_0)|+O(k^2\ell)$, $|S'| = |S|+|E|+2|\mathrm{Dom}(c_0)|+O(k^2\ell)$. Let $G_S = (V, S)$ and $G_{S'} = (V', S')$. Then $\Delta(G_{S'}) = O(\Delta(G_S) + \Delta(G) + |\mathrm{Dom}(c_0)|/\ell)$. Moreover if we are given a proper vertex p-coloring of the graph $G_S = (V, S)$ then we can output also a proper vertex $(p + 3)$-coloring of the graph $G_{S'} = (V', S')$.*

2.3 From Subset Rainbow k-Coloring to Rainbow k-Coloring

The basic idea of our reduction from Subset Rainbow k-Coloring to Rainbow k-Coloring is to modify the graph so that the pairs of vertices from $\bar{E} \setminus S$ can be somehow trivially satisfied, without affecting the satisfiability of S. To this end we use a notion of biclique covering number (called also bipartite dimension). The *biclique covering number* $\mathrm{bc}(G)$ of a graph G is the smallest number of biclique subgraphs of G that cover all edges of G. The following proposition is well-known.

▶ **Proposition 10** (Folklore). *It holds that $\mathrm{bc}(K_n) = \lceil \log n \rceil$, and the corresponding cover can be constructed in polynomial time.*

Proof. Assume $V(K_n) = \{0, \ldots, n-1\}$. The i-th biclique contains edges between the vertices that have 0 at the i-th bit and the vertices that have 1 at the i-th bit. ◀

Let $G = (V_1, V_2, E)$ be a bipartite graph. Then $\hat{G}$ denotes the bipartite complement of G, i.e, the bipartite graph $(V_1, V_2, \{v_1v_2 \ : \ v_1 \in V_1, v_2 \in V_2, \text{ and } v_1v_2 \notin E\})$. We will use the following result of Jukna. Recall that we denote $\Delta_1(G) = \max\{\Delta(G), 1\}$.

▶ **Theorem 11** (Jukna [16]). *If G is an n-vertex bipartite graph, then $\mathrm{bc}(\hat{G}) = O(\Delta_1(G) \log n)$.*

Let us call the cover from Theorem 11 the *Jukna cover*. In our application we need to be able to *compute* the Jukna cover fast.

▶ **Lemma 12.** *The Jukna cover can be constructed in (i) expected polynomial time, or (ii) deterministic $2^n n^{O(1)}$ time.*

Proof. Denote $\Delta = \Delta(G)$. If $\Delta = 0$ the claim follows from Proposition 10, so in what follows assume $\Delta \geq 1$. Jukna [16] shows a simple worst-case linear time algorithm which samples a biclique in G. Then it is proved that after sampling t bicliques, the probability that there is an edge not covered by one of the bicliques is at most $n^2e^{-t/(\Delta e)}$. It follows that the probability that more than $\Delta e(2 \ln n + 1)$ samples are needed is at most e^{-1}. If after $\Delta e(2 \ln n + 1)$ samples some edges is not covered, we discard all the bicliques found and repeat the whole algorithm from the scratch. The expected number of such restarts is $1/(1 - e^{-1}) = O(1)$.

Now we proceed to the second part of the claim. Let $G = (V_1, V_2, E)$. For every subset $A \subseteq V_1$ we define the biclique $B_A = (A, B, E_A)$, where B is the set of vertices of V_2 adjacent in $\hat{G}$ to all vertices of A. Clearly, B_A is a subgraph of $\hat{G}$ and for every subset $A \subseteq V_1$ it can be found in time linear in the size of $\hat{G}$. Our deterministic algorithm works as follows: as long as not all edges of $\hat{G}$ are covered, it picks the biclique B_A which maximizes the number of new covered edges of $\hat{G}$. Since all the bicliques in the set $\{B_A \ : \ A \subseteq V_1\}$ can be listed

in time $O(2^n|E(\hat{G})|)$, the total running time is $t2^n n^{O(1)}$, where t is the size of the returned cover. It suffices to show that $t = O(\Delta \log n)$.

Jukna [16] shows that if set A is chosen by picking every vertex of V_1 independently with probability $\frac{1}{\Delta}$, then for any edge $uv \in E(\hat{G})$, $\Pr[uv \in E_A] \geq \frac{1}{\Delta e}$. Consider any step of our algorithm and let $R \subseteq E(\hat{G})$ be the set of the edges of $\hat{G}$ which are not covered yet. By the bound on $\Pr[uv \in E_A]$ and the linearity of expectation a set A sampled as described above covers at least $|R|/(\Delta e)$ new edges in expectation. In particular, it implies that there exists a set $A \subseteq V_1$ that covers at least $|R|/(\Delta e)$ new edges. Let $\alpha = (1 - \frac{1}{\Delta e})^{-1}$. By the Taylor expansion of $\log(1-x)$, it follows that $t = O(\log_\alpha |E(\hat{G})|) = O(\log n / \log \alpha) = O(\Delta \log n)$. ◀

▶ **Lemma 13.** *Let G be an n-vertex graph with a given proper vertex p-coloring. Then the edges of $\bar{G}$ can be covered by $O(p^2\Delta_1(G)\log n)$ bicliques from $\bar{G}$ so that any edge of G and any biclique have at most one common vertex. This cover can be constructed in (i) expected polynomial time, or (ii) deterministic $2^n n^{O(1)}$ time.*

Proof. The edges of $\bar{G}$ between the vertices of any color class form a clique, so by Proposition 10 we can cover its edges using $O(\log n)$ bicliques. If an edge of G has both endpoints in such a biclique, these endpoints have the same color, contradiction. For two different colors i and j the edges of G between their color classes form a bipartite graph of maximum degree at most $\Delta(G)$. Hence by Lemma 12 we can cover the edges of its bipartite complement using $O(\Delta_1(G)\log n)$ bicliques. If an edge uv of G has both endpoints in such a biclique, then either (i) these endpoints have the same color, contradiction, or (ii) these endpoints belong to two different parts of the biclique, so uv is in the biclique and hence $uv \in E(\bar{G})$, a contradiction. Summing over all color classes and pairs of color classes, we use $O(p^2\Delta_1(G)\log n)$ bicliques, as required. ◀

Now we proceed to the actual reduction.

▶ **Lemma 14.** *Given an instance $(G = (V, E), S)$ of* Subset Rainbow 2-Coloring *together with a proper p-coloring of the graph $G_S = (V, S)$, one can construct an equivalent instance G' of* Rainbow 2-Coloring *such that $|V(G')| = O(|V| + kp^2\Delta_1(G_S)\log|V|)$, $|E(G')| = O(|E(G)|+(|V|+p^2\Delta_1(G_S)\log|V|)\cdot p^2\Delta_1(G_S)\log|V|)$. The construction algorithm can run in (i) expected polynomial time, or (ii) deterministic $2^{|V|}|V|^{O(1)}$ time.*

Proof. Here we focus on the $k = 2$ case. The $k \geq 3$ case is significantly more technical — see the details in the full version. Let us consider a biclique covering of the complement of the graph G_S with $q = O(p^2\Delta_1(G_S)\log n)$ bicliques $(U_1, V_1; E_1), (U_2, V_2; E_2), \ldots, (U_q, V_q; E_q)$ as in Lemma 13. Let $W = \{w_1, w_2, \ldots, w_q\}$, $T = \{t_1, t_2, t_3\}$, $V(G') = V \cup W \cup T$ and $E(G') = E(G) \cup (W \times W) \cup (T \times T) \cup (\{t_2\} \times W) \cup (\{t_3\} \times (V \cup W)) \cup \left(\bigcup_{1 \leq i \leq q} \{w_i\} \times (U_i \cup V_i)\right)$ (we abuse the notation assuming that $\times$ operator returns *unordered* pairs minus loops). Because of the space limitation the equivalence proof is deferred to the full version. ◀

2.4 Putting everything together

By pipelining lemmas 7, 8, and 9 we get the following corollary.

▶ **Corollary 15.** *Fix $k \geq 2$. Given a* 3-SAT *formula φ with m clauses one can construct in polynomial time an equivalent instance $(G = (V, E), S)$ of* Subset Rainbow k-Coloring *such that $|V| = O(m^{2/3})$, $|E| = O(m)$, $\Delta((V, S)) = O(m^{1/3})$, and the graph $G_S = (V, S)$ is $O(1)$-colorable.*

Note that in Corollary 15 we have $|S| = |V|\Delta((V,S)) = O(m)$. It follows that the Sparsification Lemma (Lemma 6) and Corollary 15 imply Theorem 3.

Pipelining Corollary 15 and Lemma 14 gives the following corollary.

▶ **Corollary 16.** *Fix $k \geq 2$. Given a* 3-SAT *formula φ with $O(m)$ clauses one can construct an equivalent instance G of* Rainbow k-Coloring *with $O(m^{2/3})$ vertices and $O(m \log m)$ edges. The construction algorithm can run in (i) expected polynomial time, or (ii) deterministic $2^{O(m^{2/3})}$ time.*

Again, the above and the Sparsification Lemma immediately imply Theorem 2.

3 Algorithms for Subset Rainbow k-Coloring

In this section we study FPT algorithms for Subset Rainbow k-Coloring parameterized by $|S|$. We provide two such algorithms, based on different approaches: one for $k = 2$ case, and one (slightly slower) for the general case. Consider an instance (G, S) of the Subset Rainbow k-Coloring problem. Note that we can assume that $S \subseteq \bar{E}$, since any constraint $\{u, v\} \in E$ is satisfied in every edge coloring. Moreover, we say that a pair $\{u, v\}$ is *feasible* when the distance between u and v is at most k. The set of all feasible pairs is denoted by $F(G)$. Clearly, when S contains a request which is not feasible, then (G, S) is a trivial NO-instance. Hence, throughout this section we assume $S \subseteq \bar{E} \cap F(G)$.

3.1 The $k = 2$ case

For any $X \subseteq S$ let $\mathcal{P}_X$ be the set of all 2-edge paths between the pairs of vertices in X. Denote $E(\mathcal{P}_X) = \bigcup_{P \in \mathcal{P}_X} E(P)$. For two edges $e_1, e_2 \in E(G)$ we say that e_1 and e_2 are *linked by X*, denoted as $e_1 \sim_X e_2$ when there are two paths $P_1, P_2 \in \mathcal{P}_X$ (possibly $P_1 = P_2$) such that $e_1 \in E(P_1)$, $e_2 \in E(P_2)$ and $E(P_1) \cap E(P_2) \neq \emptyset$. Let $\approx_X$ be the transitive closure of $\sim_X$. Then $\approx_X$ is an equivalence relation. Recall that $E(G)/\approx_X$ denotes the quotient set of the relation $\approx_X$. The main observation of this section is the following theorem.

▶ **Theorem 17.** *The number of 2-colorings of $E(G)$ that satisfy all the pairs in S is equal to $\sum_{X \subseteq S} (-1)^{|X|} 2^{|E(G)/\approx_X|}$.*

In the proof we make use of the well-known inclusion-exclusion principle. Below we state it in the intersection version (see, e.g., [10])

▶ **Theorem 18** (Inclusion–exclusion principle, intersection version)**.** *Let $A_1, \ldots, A_n \subseteq U$, where U is a finite set. Denote $\bigcap_{i \in \emptyset}(U \setminus A_i) = U$. Then*

$$\Big|\bigcap_{i \in [n]} A_i\Big| = \sum_{X \subseteq [n]} (-1)^{|X|} \Big|\bigcap_{i \in X} (U \setminus A_i)\Big|.$$

Proof of Theorem 17. Let us define, for every pair $\{u, v\} \in S$ (say, $u < v$), the set $A_{u,v}$ of 2-edge colorings of G that satisfy $\{u, v\}$. Note that the number of rainbow 2-colorings of G that satisfy all the pairs in S is equal to $|\bigcap_{\{u,v\} \in S} A_{u,v}|$. By Theorem 18 it suffices to show that, for any subset $X \subseteq S$, the number $\#_X$ of 2-colorings such that *none* of the pairs in X is satisfied, equals $2^{|E(G)/\approx_X|}$.

Fix any coloring c that does not satisfy any pair from X. Then every path from $\mathcal{P}_X$ has both edges of the same color. Hence, for two edges $e_1, e_2 \in E(G)$, if $e_1 \sim_X e_2$ then e_1 and e_2 are colored by c with the same color. It follows that for any equivalence class A of $\approx_X$, all edges of A are have the same color in c. This proves that $\#_X \leq 2^{|E(G)/\approx_X|}$.

For every function $c_0 : (E(G)/\approx_X) \to \{1,2\}$ we can define the coloring $c : E(G) \to \{1,2\}$ by putting $c(e) = c_0([e]_{\approx_X})$ for every edge $e \in E(G)$. (Note that the edges that do not belong to any path in $\mathcal{P}_X$ form singleton equivalence classes.) Then, c does not satisfy any pair from X, because if some pair $\{u,v\}$ is satisfied then there is a 2-color path uxv; but $ux \sim_X xv$, so $[ux]_{\approx_X} = [xv]_{\approx_X}$ and $c(ux) = c(xv)$, a contradiction. It follows that $\#_X \geq 2^{|E(G)/\approx_X|}$. ◀

Since it is a standard exercise to compute the relation $X \subseteq S$ in $O(|E| + |S| \cdot |V|)$ time (see the full version), we get the following corollary. (Let us remark here that the algorithm from Corollary 19 only decides whether the coloring exists, without finding it. However, by a minor modification of the algorithm it can construct the coloring; see the full version.)

▶ **Corollary 19.** *For any graph $G = (V,E)$ and a set of requests S the number of 2-colorings of E that satisfy all the pairs in S can be computed in $O(2^{|S|}(|E| + |S| \cdot |V|))$ time and polynomial space. In particular,* Subset Rainbow 2-Coloring *can be decided within the same time.*

3.2 The general case

In this section we use partial colorings. For convenience, a partial coloring is represented as a function $c : E \to [k] \cup \{\perp\}$, where the value $\perp$ corresponds to an uncolored edge. By $\mathrm{Dom}(c)$ we denote the domain of the corresponding partial function, i.e., $\mathrm{Dom}(c) = c^{-1}([k])$. The partial coloring which does not color anything, i.e., is constantly equal to $\perp$ is denoted by $c_\perp$.

For a graph $G = (V,E)$ consider a partial edge coloring $c : E \to [k] \cup \{\perp\}$. A *guide function* is any function of the form $f : S \to \binom{\mathrm{Dom}(c)}{\leq k}$, i.e., any function that assigns sets of at most k colored edges to all requests in S. A constant guide function equal to $\emptyset$ for every request in S is denoted by $g_{S,\emptyset}$. Pick any pair $\{u,v\} \in S$. We say that a walk W connecting u and v is f-guided if every color appears at most once on W, and $f(\{u,v\}) \subseteq E(W)$. We say that a coloring c is (f,S)-rainbow when for every pair $\{u,v\} \in S$ there is an f-guided walk between u and v. Note that (G,S) is a YES-instance of Subset Rainbow k-Coloring iff there is an $(g_{S,\emptyset}, S)$-rainbow coloring. Indeed, every rainbow walk contains a rainbow path.

The following lemma is going to be useful in our branching algorithm.

▶ **Lemma 20.** *Let $G = (V,E)$ be a graph, and let S be a set of requests. Let $c_0 : E \to [k]$ be a partial edge coloring and let $f : S \to \binom{\mathrm{Dom}(c_0)}{\leq k}$ be a guide function. Then, given a pair $\{u,v\} \in S$ in time $2^k n^{O(1)}$ one can find an f-guided u-v walk of length at most k, if it exists.*

Proof. The algorithm is as follows. We can assume that $f(\{u,v\})$ does not contain two edges of the same color, for otherwise the requested walk does not exist. For every $e \in f(\{u,v\})$ we remove all the edges of color $c_0(e)$. Next, we put back edges of $f(\{u,v\})$. Then it suffices to find in the resulting graph G' any u-v path of length at most k and with no repeated colors that visits all the colors of the edges in $f(\{u,v\})$. This is done using dynamic programming. For every vertex $x \in V$, subset $X \subseteq [k]$ and integer $\ell = 0, \ldots, k$ we find the boolean value $T[x,X,\ell]$ which is true iff there is a u-x walk of length ℓ which does not repeat colors and visits all the colors from X, but not more. We initialize $T[u,\emptyset,0] = \texttt{true}$ and $T[x,\emptyset,0] = \texttt{false}$ for every $x \neq u$. Next we iterate through the remaining triples (x,X,ℓ), in the nondecreasing order of ℓ and X's cardinalities. The value of $T[x,X,\ell]$ is then computed using the formula

$$T[x,X,\ell] = \bigvee_{yx \in c_0^{-1}(X \cup \{\perp\}) \cap E(G')} T[y, X \setminus \{c_0(yx)\}, \ell - 1].$$

Pseudocode 1: FindColoring(S_0, c_0, f)

```
1  if S_0 = ∅ then
2      return c_0
3  if for some r ∈ S_0 there are edges e_1, e_2 ∈ f(r) with c_0(e_1) = c_0(e_2) then
4      return null
5  Pick any {u, v} ∈ S_0;
6  Find any f-guided u-v walk W of length at most k using Lemma 20;
7  if W does not exist then
8      return null
9  Let c_1 be obtained from c_0 by coloring the uncolored edges of W to get a rainbow walk;
10 if FindColoring(S_0 \ {u, v}, c_1, f|_{S_0 \ {u,v}}) ≠ null then return the coloring found;
11 for e ∈ E(W) \ Dom(c_0) do
12     for α ∈ [k] do
13         for r ∈ S_0 \ {{u, v}} do
14             Let c_{e,α} be obtained from c_0 by coloring e with α;
15             Let f_{e,r} be obtained from f by putting f(r) := f(r) ∪ {e};
16             if FindColoring(S_0, c_{e,α}, f_{e,r}) ≠ null then return the coloring found;
17 return null
```

The requested walk exists iff $T[v, X, \ell] = \texttt{true}$ for any $\ell = 0, \ldots, k$ and X such that $c_0(f(\{u, v\})) \subseteq X$. The walk is retrieved using standard DP methods. ◀

Now we are ready to describe our branching algorithm. Let $(G = (V, E), S)$ be the input instance. Our algorithm consists of a recursive procedure FindColoring which gets three parameters: S_0 (a set of requests), $c_0 : E \to [k] \cup \{\perp\}$ (a partial coloring), and a guide function $f : S \to \binom{\mathrm{Dom}(c_0)}{\leq k}$. It is assumed that for every request $r \in S$, every pair of different edges $e_1, e_2 \in f(r)$ is colored differently by c_0. The goal of the procedure FindColoring is to find an (f, S_0)-rainbow coloring $c : E \to [k]$ which extends c_0. Thus the whole problem is solved by invoking FindColoring$(S, c_\perp, g_{S,\emptyset})$. A rough description of FindColoring is as follows. We pick any pair $\{u, v\} \in S_0$ and we find any f-guided u-v walk W of length at most k using Lemma 20. Let c_1 be obtained from c_0 by coloring the uncolored edges of W to get a rainbow walk. If FindColoring$(S_0 \setminus \{u, v\}, c_1, f|_{S_0 \setminus \{u,v\}})$ returns a coloring, we are done. But if no such coloring exists then we know that we made a wrong decision: coloring some of the uncolored edges e of W into $c_1(e)$ (instead of some color α) makes some other request $r \in S_0 \setminus \{\{u, v\}\}$ impossible to satisfy. For every possible triple (e, α, r) we invoke FindColoring with the same set of requests S_0, partial coloring c_0 extended by coloring e with α, and the guide function f extended by putting $f(r) := f(r) \cup \{e\}$.

A precise description of procedure FindColoring can be found in Pseudocode 1. The following lemma proves its correctness.

▶ **Lemma 21.** *Procedure* FindColoring *invoked with parameters* (S_0, c_0, f) *finds an* (f, S_0)*-rainbow coloring* $c : E \to [k]$ *which extends* c_0*, whenever it exists.*

Proof. The proof is by induction on the sum of $|S_0|$ and the number of uncolored edges. It is clear that if $|S_0| = 0$ or all the edges are colored then the algorithm behaves correctly. In the induction step, the only non-trivial thing to check is whether any of the calls in lines 10 or 16 returns a coloring, provided that there is a solution, i.e., an (f, S_0)-rainbow coloring $c : E \to [k]$ which extends c_0. Assume that no coloring is returned in Line 16. Then for every edge $e \in E(W) \setminus \mathrm{Dom}(c_0)$, and request $r \in S_0 \setminus \{\{u, v\}\}$ coloring c is not a $(f_{e,r}, S_0)$-rainbow coloring, for otherwise the call FindColoring$(S_0, c_{e,c(e)}, f_{e,r})$ returns a coloring. If follows that for every edge $e \in E(W) \setminus \mathrm{Dom}(c_0)$ and request $r \in S_0 \setminus \{\{u, v\}\}$ the walk that realizes

the request r in the coloring c does not contain e. Hence, the following coloring

$$c'(e) = \begin{cases} c(e) & \text{if } e \notin E(W), \\ c_1(e) & \text{if } e \in E(W). \end{cases}$$

is another (f, S_0)-rainbow coloring, and it extends c_1. It follows that the call in Line 10 returns a coloring, as required. ◀

▶ **Theorem 22.** *For every integer k, there is an FPT algorithm for* Subset Rainbow k-Coloring *parameterized by $|S|$. The algorithm runs in time $(k^2|S|)^{k|S|}2^k n^{O}(1)$, in particular in $|S|^{O(|S|)}n^{O(1)}$ time for every fixed k.*

Proof. By Lemma 21 Subset Rainbow k-Coloring is solved by invoking FindColoring$(S, c_\perp, g_{S,\emptyset})$. Note that whenever we go deeper in the recursion either some request of S_0 gets satisfied, or $|f(r)|$ increases for some $r \in S_0$. When $|f(r)|$ increases to $k+1$, the corresponding recursive call returns `null` immediately (because the condition in Line 3 holds). It follows that the depth of the recursion is at most $|S|k$. Since in every call of Subset Rainbow k-Coloring the algorithm uses time $2^k n^{O(1)}$ (by Lemma 21) and branches into at most $1 + k^2(|S| - 1) \leq k^2|S|$ recursive calls, the total time is $(k^2|S|)^{k|S|}2^k n^{O}(1)$, as required. ◀

4 Further Work

We believe that this work only initiates the study of fine-grained complexity of variants of Rainbow k-Coloring. In particular, many open questions are still unanswered. The ultimate goal is certainly to get tight bounds. We pose the following two conjectures.

▶ **Conjecture 23.** *For any integer $k \geq 2$, there is no $2^{o(|E|)}n^{O(1)}$-time algorithm for* Rainbow k-Coloring*, unless ETH fails.*

▶ **Conjecture 24.** *For any integer $k \geq 2$, there is no $2^{o(n^2)}n^{O(1)}$-time algorithm for* Rainbow k-Coloring*, unless ETH fails.*

Note that in this work we have settled Conjecture 23 for Subset Rainbow k-Coloring, and for Rainbow k-Coloring we showed a slightly weaker, $2^{o(|E|/\log|E|)}n^{O(1)}$ bound. However, avoiding this $\log|E|$ factor seems to constitute a considerable technical challenge.

In this paper we gave two algorithms for Subset Rainbow k-Coloring parameterized by $|S|$, one working in $2^{|S|}n^{O(1)}$ time for $k = 2$ and another, working in time $|S|^{O(|S|)}n^{O(1)}$ for every fixed k. We conjecture that there exists an algorithm running in time $2^{O(|S|)}n^{O(1)}$ for every fixed k.

Finally, we would like to propose yet another parameterization of Rainbow k-Coloring. Assume we are given a graph $G = (V, E)$ and a subset of vertices $S \subseteq V$. In the Steiner Rainbow k-Coloring problem the goal is to determine whether there is a rainbow k-coloring such that every pair of vertices in S is connected by a rainbow path. By our Theorem 2, Steiner Rainbow k-Coloring has no algorithm running in time $2^{o(|S|^{3/2})}$, under ETH. On the other hand, our algorithm for Subset Rainbow k-Coloring implies that Steiner Rainbow k-Coloring parameterized by $|S|$ admits an FPT algorithm with running time of $2^{O(|S|^2\log|S|)}n^{O(1)}$. It would be interesting make the gap between these bounds smaller.

References

1 Prabhanjan Ananth, Meghana Nasre, and Kanthi K. Sarpatwar. Rainbow connectivity: Hardness and tractability. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2011)*, pages 241–251, 2011.

2 Yair Caro, Arie Lev, Yehuda Roditty, Zsolt Tuza, and Raphael Yuster. On rainbow connection. *Electron. J. Combin*, 15(1):R57, 2008.

3 Sourav Chakraborty, Eldar Fischer, Arie Matsliah, and Raphael Yuster. Hardness and algorithms for rainbow connection. *J. of Combinatorial Optimization*, 21(3):330–347, 2009.

4 L. Sunil Chandran and Deepak Rajendraprasad. Rainbow Colouring of Split and Threshold Graphs. *Computing and Combinatorics*, pages 181–192, 2012.

5 L. Sunil Chandran and Deepak Rajendraprasad. Inapproximability of rainbow colouring. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2013)*, pages 153–162, 2013.

6 L. Sunil Chandran, Deepak Rajendraprasad, and Marek Tesař. Rainbow colouring of split graphs. *Discrete Applied Mathematics*, 2015. doi:10.1016/j.dam.2015.05.021.

7 Gary Chartrand, Garry L. Johns, Kathleen A. McKeon, and Ping Zhang. Rainbow connection in graphs. *Mathematica Bohemica*, 133(1), 2008.

8 Gary Chartrand and Ping Zhang. *Chromatic graph theory*. CRC press, 2008.

9 Marek Cygan, Fedor V. Fomin, Alexander Golovnev, Alexander S. Kulikov, Ivan Mihajlin, Jakub Pachocki, and Arkadiusz Socała. Tight bounds for graph homomorphism and subgraph isomorphism. In *Proc. of the 27th Annual ACM-SIAM Symp. on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1643–1649, 2016.

10 Marek Cygan, Fedor V Fomin, Łukasz Kowalik, Dániel Lokshtanov, Daniel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.

11 Reinhard Diestel. *Graph Theory*. Springer-Verlag Heidelberg, 2010.

12 Eduard Eiben, Robert Ganian, and Juho Lauri. On the complexity of rainbow coloring problems. In *Proceedings of the Twenty-Sixth International Workshop on Combinatorial Algorithms, IWOCA 2015, Verona, Italy, October 5-7*, pages 209–220, 2015. URL: http://arxiv.org/abs/1510.03614, doi:10.1007/978-3-319-29516-9_18.

13 Michael Held and Richard M. Karp. A dynamic programming approach to sequencing problems. *Journal of SIAM*, 10:196–210, 1962.

14 Russell Impagliazzo and Ramamohan Paturi. On the Complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001. doi:10.1006/jcss.2000.1727.

15 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.

16 Stasys Jukna. On set intersection representations of graphs. *Journal of Graph Theory*, 61(1):55–75, 2009. doi:10.1002/jgt.20367.

17 Lukasz Kowalik, Juho Lauri, and Arkadiusz Socala. On the fine-grained complexity of rainbow coloring. *CoRR*, abs/1602.05608, 2016. URL: http://arxiv.org/abs/1602.05608.

18 Eugene L. Lawler. A note on the complexity of the chromatic number problem. *Information Processing Letters*, 5(3):66–67, 1976.

19 Van Bang Le and Zsolt Tuza. Finding optimal rainbow connection is hard. Technical Report CS-03-09, Universität Rostock, 2009.

20 Xueliang Li, Yongtang Shi, and Yuefang Sun. Rainbow Connections of Graphs: A Survey. *Graphs and Combinatorics*, 29(1):1–38, 2012.

21 Arkadiusz Socała. Tight lower bound for the channel assignment problem. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 662–675, 2015.

22 Craig A. Tovey. A simplified NP-complete satisfiability problem. *Discrete Applied Mathematics*, 8(1):85–89, 1984. doi:10.1016/0166-218X(84)90081-7.

23 Kei Uchizawa, Takanori Aoki, Takehiro Ito, Akira Suzuki, and Xiao Zhou. On the Rainbow Connectivity of Graphs: Complexity and FPT Algorithms. *Algorithmica*, 67(2):161–179, 2013.