The Parameterized Complexity of the Rainbow Subgraph Problem

Falk Hüffner*, Christian Komusiewicz**, Rolf Niedermeier, and Martin Rötzschke

Institut für Softwaretechnik und Theoretische Informatik, TU Berlin, Germany
{falk.hueffner,christian.komusiewicz,rolf.niedermeier}@tu-berlin.de

Abstract. The NP-hard Rainbow Subgraph problem, motivated from bioinformatics, is to find in an edge-colored graph a subgraph that contains each edge color exactly once and has at most k vertices. We examine the parameterized complexity of Rainbow Subgraph for paths, trees, and general graphs. We show, for example, APX-hardness even if the input graph is a properly edge-colored path in which every color occurs at most twice. Moreover, we show that Rainbow Subgraph is W[1]-hard with respect to the parameter k and also with respect to the dual parameter $\ell := n - k$ where n is the number of vertices. Hence, we examine parameter combinations and show, for example, a polynomial-size problem kernel for the combined parameter ℓ and "maximum number of colors incident with any vertex".

1 Introduction

The Rainbow Subgraph problem is defined as follows.

> Rainbow Subgraph
> **Instance:** An undirected graph $G = (V, E)$, an edge coloring $\mathrm{col} : E \to \{1, \ldots, p\}$ for some $p \geq 1$, and an integer $k \geq 0$.
> **Question:** Is there a subgraph G' of G that contains each edge color exactly once and has at most k vertices?

We call a subgraph G' with these properties a *solution* of order at most k. In the problem name, the term *rainbow* refers to the fact that all edges of G' have a different color. For convenience, we define a *rainbow cover* as a subgraph where every color occurs at least once. Note that every rainbow cover G' of order at most k has a subgraph that is a solution: Simply remove any edge whose color appears more than once in G'. Repeating this operation as long as possible yields a solution of the same order as G'.

Rainbow Subgraph arises in bioinformatics: The (Population) Parsimony Haplotyping problem can be reduced to Rainbow Subgraph [11]; note, however, that depending on the input, this reduction might not produce a polynomial-size instance. Another bioinformatics application appears in the context of PCR primer set design for spotted microarray experiments [5].

* Supported by DFG project ALEPH (HU 2139/1).
** Partially supported by a post-doctorial grant funded by the Région Pays de la Loire.

To appear in *Proceedings of the 40th International Workshop on Graph-Theoretic Concepts in Computer Science (WG '14)*, Lecture Notes in Computer Science, Springer, 2014.

Table 1. Complexity overview for Rainbow Subgraph. The $O^*()$-notation suppresses factors polynomial in the input size.

Par.	Paths	Trees	General graphs
p	$O^*(3^p)$ (Thm. 4)	$O^*(3^p)$ (Thm. 4)	W[1]-hard (Thm. 2)
p, Δ	— " —	— " —	$O^*((4\Delta-4)^p)$ (Thm. 3)
k	$O^*(3^k)$ (Thm. 4+(2))	$O^*(3^k)$ (Thm. 4+(3))	W[1]-hard (Thm. 2+(1))
k, Δ	— " —	— " —	$O^*(2^{k\Delta/2})$ (Thm. 3)
ℓ	$O^*(5^\ell)$ (Thm. 8)	W[1]-hard (Thm. 5)	W[1]-hard (Thm. 5)
ℓ, Δ_C	— " —	$O^*((2\Delta_C+1)^\ell)$ (Thm. 8)	$O^*((2\Delta_C+1)^\ell)$ (Thm. 8)
			$O(\Delta_C^3\ell^4)$-vertex kernel (Thm. 7)
ℓ, q	— " —	W[1]-hard (Thm. 5)	W[1]-hard (Thm. 5)
Δ, q	APX-hard (Thm. 1)	APX-hard (Thm. 1)	APX-hard ([8])

Previous work. The optimization version of Rainbow Subgraph has been mostly studied in terms of polynomial-time approximability. Here the optimization goal is to minimize the number of vertices in the solution; we refer to this problem as Minimum Rainbow Subgraph. Minimum Rainbow Subgraph is APX-hard even on graphs with maximum vertex degree $\Delta \geq 2$ in which every color occurs at most twice [8]. Moreover, Minimum Rainbow Subgraph cannot be approximated within a factor of $c \ln \Delta$ for some constant c unless NP has slightly superpolynomial time algorithms [12].

The more general Minimum-Weight Multicolored Subgraph problem has a randomized $\sqrt{q \log p}$-approximation algorithm, where q is the maximum number of times any color occurs in the input graph [7]. Minimum Rainbow Subgraph can be approximated within a ratio of $(\delta + \ln\lceil\delta\rceil + 1)/2$, where δ is the average vertex degree in the solution [9], and within a factor of $\max(\sqrt{2n}, \sqrt{\Delta}(1 + \sqrt{\ln \Delta/2}))$ [12]. Katrenič and Schiermeyer [8] present an exact algorithm for Rainbow Subgraph that has a running time of $n^{O(1)} \cdot 2^p \cdot \Delta^{2p}$, where Δ is the maximum vertex degree of the input.

Our contributions. Since Rainbow Subgraph is NP-hard even on collections of paths and cycles [8], we perform a broad parameterized complexity analysis. Table 1 gives an overview on the complexity of Minimum Rainbow Subgraph on paths, trees, and general graphs, when parameterized by

- p: number of colors;
- k: number of vertices in the solution;
- $\ell := n - k$: number of vertex deletions to obtain a solution;
- Δ: maximum vertex degree;
- $\Delta_C := \max_{v \in V} |\{c \mid \exists \{u, v\} \in E : \mathrm{col}(\{u, v\}) = c\}|$: maximum color degree;
- q: maximum number of times any color occurs in the input graph.

For each parameter and some parameter combinations, we give either a fixed-parameter algorithm, show W[1]-hardness, or show NP-hardness for constant parameter values.

Our main results are as follows: Rainbow Subgraph is APX-hard even if the input graph is a properly edge-colored path with $q = 2$. Rainbow Subgraph is W[1]-hard on general graphs for each of the considered parameters. For the number of colors p, solution order k, and number ℓ of vertex deletions, the complexity seems to depend on the density of the graph as the problem is W[1]-hard for each of these parameters but it becomes tractable if any of these parameters is combined with the maximum degree Δ. For the parameter ℓ, W[1]-hardness holds even if the input graph is a tree.

Preliminaries. APX is the class of optimization problems that allow constant-factor approximations. If a problem is APX-hard, then it cannot be approximated in polynomial time to arbitrary constant factors, unless P = NP. A problem is called *fixed-parameter tractable* (FPT) with respect to some problem-specific parameter x if it can be solved in $f(x) \cdot |I|^{O(1)}$ time, where $|I|$ is the instance size and f is an arbitrary computable function. A *kernel* for a parameterized problem is, roughly, a polynomial-time self-reduction that results in an instance whose size is bounded only in the parameter. Analogously to NP, the class W[1] captures parameterized hardness. It is widely assumed that if a problem is W[1]-hard, then it is not fixed-parameter tractable.

We will use the following simple observation several times.

Observation 1. *Let $G' = (V', E')$ be a solution for a* Rainbow Subgraph *instance with $G = (V, E)$. If there are two vertices u, v in V' such that $\{u, v\} \in E$ but $\{u, v\} \notin E'$, then there is a solution G'' that does contain the edge $\{u, v\}$ and has the same number of vertices.*

Observation 1 is true since replacing the edge in G' that has the same color as $\{u, v\}$ by $\{u, v\}$ is a solution. Next, we list some easy to see observations regarding parameter bounds:

$$p \leq k(k-1)/2, \tag{1}$$

$$p \leq k\Delta/2, \tag{2}$$

$$p \leq k - 1 \qquad \text{if } G \text{ is acyclic.} \tag{3}$$

Due to lack of space, some proofs are deferred to a long version of this article.

2 Parameterization by Color Occurrences

We now consider the complexity of Rainbow Subgraph parameterized by the maximum number of color occurrences q. Indeed, the value q is bounded in some applications: For example in the graph formulation of Parsimony Haplotyping, q depends on the maximum number of ambiguous positions in a genotype which can be assumed to be small. Unfortunately, Rainbow Subgraph remains hard under q-parameterization, even for heavily restricted graph classes.

Katrenič and Schiermeyer [8] showed that Minimum Rainbow Subgraph is APX-hard for $\Delta = 2$. The instances produced by their reduction contain

precisely two edges of each color, so APX-hardness even holds for $q = 2$. However, the resulting graph contains cycles and is not properly edge-colored, so the complexity on acyclic graphs and on properly edge-colored graphs (like those resulting from Parsimony Haplotyping instances) remains to be explored. We show that neither restriction is helpful as Rainbow Subgraph is APX-hard for properly edge-colored paths with $q = 2$. This strengthens the hardness result of Katrenič and Schiermeyer [8]. For this purpose, we develop an L-reduction from the following special case of Minimum Vertex Cover:

Minimum Vertex Cover in Cubic Graphs
Instance: An undirected graph $H = (W, F)$ in which every vertex has degree three.
Task: Find a minimum-cardinality vertex cover of G.

Minimum Vertex Cover in Cubic Graphs is APX-complete [1].

Theorem 1. Minimum Rainbow Subgraph *is APX-hard even when the input is a properly edge-colored path in which every color occurs at most twice.*

Proof. Given an instance $H = (W = \{w_1, \ldots, w_n\}, F)$ of Minimum Vertex Cover in Cubic Graphs, construct an edge-colored path $G = (V, E)$ as follows. The vertex set is $V := \{v_1, \ldots, v_{16n+2}\}$. The edge set is $E := \{\{v_i, v_{i+1}\} \mid 1 \leq i \leq 16n + 1\}$, that is, vertices with successive indices are adjacent. It remains to specify the edge colors. Herein, we use u^* to denote *unique* colors, that is, if an edge is u^*-colored, then it receives an edge color that does not appear anywhere else in G. In addition to these unique colors, introduce five colors for each vertex of H, that is, for each $w_i \in W$ create edge colors c_i, c_i', c_i'', x_i, and y_i. The colors c_i, c_i', and c_i'' are "filling" colors which are needed because G is connected. Furthermore, for each edge $f_i \in F$ introduce an edge color ϕ_i.

Now, color the first $6n + 1$ edges of G by the sequence

$$u^*\, c_1\, u^*\, c_1'\, u^*\, c_1''\, u^*\, c_2\, u^*\, c_2'\, u^*\, c_2''\, u^*\, \cdots\, c_2\, u^*\, c_2'\, u^*\, c_2''\, u^*.$$

That is, the edge between v_0 and v_1 is u^*-colored, the edge between v_1 and v_2 receives color c_1, and so on. The u^*-colors are unique and thus occur only once in G. Thus, both endpoints of these colors are contained in every solution.

Now for each vertex w_i in H color 10 edges in G according to the edges that are incident with w_i. More precisely, for each w_i color the edges from $v_{6n+2+10(i-1)}$ to $v_{6n+2+10i}$. We call the subpath of G with these vertices the w_i-*part* of G. Let $\{f_r, f_s, f_r\}$ denote the set of edges incident with w_i. Then color the edges between $v_{6n+2+10(i-1)}$ and $v_{6n+2+10i}$ by the sequence

$$c_i\, \phi_r\, x_i\, \phi_s\, c_i'\, y_i\, \phi_t\, c_i''\, x_i\, y_i.$$

That is, the edge between $v_{6n+2+10(i-1)}$ and $v_{6n+2+10(i-1)+1}$ receives color c_i, the edge between $v_{6n+2+10(i-1)+1}$ and $v_{6n+2+10(i-1)+2}$ receives color ϕ_r, and so on. The resulting graph is a path with exactly $16 \cdot n + 1$ edges and $p = 8 \cdot n + |F| + 1$ colors.

The idea of the construction is that we may use the vertices of the w_i-part to "cover" the colors corresponding to the edges incident with w_i. If we do so, then the solution has two connected components in the w_i-part. Otherwise, it is sufficient to include one connected component from the w_i-part. Since the solution graph is acyclic and the number of edges in a minimal solution is fixed, the number of connected components in the solution and its order are equal up to an additive constant.

We now show formally that the reduction fulfills the two properties of L-reductions [14]. Let S^* be an optimal vertex cover for the Minimum Vertex Cover in Cubic Graphs instance and let G^* be an optimal solution to the constructed Minimum Rainbow Subgraph instance.

The first property we need to show is that $|V(G^*)| = O(|S^*|)$. As observed above, the number of colors p in G is $O(n + |F|)$ and thus $|V(G^*)| \leq 2p = O(n + |F|)$. Clearly, S^* contains at least $|F|/3$ vertices, since every vertex in H covers at most three edges. Moreover, since H is cubic we have $n < 2|F|$ and thus $|S^*| = \Theta(n + |F|)$. Consequently, $|V(G^*)| = O(|S^*|)$.

The second property we need to show is following: given a solution G' to G, we can compute in polynomial time a solution S' to G such that

$$|S'| - |S^*| = O(|V(G')| - |V(G^*)|).$$

Let G' be a solution to G. The proof outline is as follows. We show that G' has order at least $p + n + 1 + x$, $x \geq 0$, and that, given G', we can compute in polynomial time a size-x vertex cover S' of H. Then we show that, conversely, there is a solution of order at most $p + n + 1 + |S^*|$. Thus, the differences between the solution sizes in the Minimum Vertex Cover in Cubic Graphs instance and in the Minimum Rainbow Subgraph instance are essentially the same. We omit the details. □

3 Parameterization by Number of Colors

We now consider the parameter number of colors p. We show that Rainbow Subgraph is generally W[1]-hard with respect to p but becomes fixed-parameter tractable if the input graph is sparse. By Eq. (1), and the fact that we can always construct a solution by arbitrarily selecting one edge of each color, implying $k \leq 2p$, the parameter p is polynomially upper- and lower-bounded by the solution order k. In consequence, while our main focus is on parameter p, every parameterized complexity result for p also implies the corresponding parameterized complexity result for k.

A graph G is called *d-degenerate* if every subgraph of G has a vertex of degree at most d. We can show that even on 2-degenerate bipartite graphs, the decision problem Rainbow Subgraph is W[1]-hard for parameter p (and thus also for parameter k) by a parameterized reduction from the Multicolored Clique problem.

Theorem 2. Minimum Rainbow Subgraph *is W[1]-hard with respect to the number of colors p, even if the input graph is 2-degenerate and bipartite.*

Replacing degeneracy by the larger parameter maximum degree Δ of G yields fixed-parameter tractability: Katrenič and Schiermeyer [8] proposed an algorithm that solves Minimum Rainbow Subgraph in $(2\Delta^2)^p \cdot n^{O(1)}$ time. We show an improved bound:

Theorem 3. *Let (G, col) be an instance of* Minimum Rainbow Subgraph *with p colors and maximum vertex degree Δ. An optimal solution can be computed in $O((4\Delta-4)^p \cdot \Delta n^2)$ time or in $O((4\Delta-4)^k \cdot n^2 + 2^{k\Delta/2} \cdot (k\Delta)^3 \log(k\Delta))$ time, where k is the order of the solution.*

To prove Theorem 3, we follow a two-step approach: First, we enumerate connected candidate subgraphs exploiting the sparseness constraint. Second, we select from these candidate subgraphs a minimum-order set with all colors, exploiting techniques by Björklund et al. [2].

The algorithm by Katrenič and Schiermeyer [8] has a somewhat different structure, but can also be understood in terms of a subgraph enumeration process and a combinatorial part: It employs a method for enumerating all connected rainbow subgraphs in $O(\Delta^{2p} \cdot np)$ time and finds a solution via dynamic programming. In contrast, we consider only connected *induced* subgraphs in the first step, which improves efficiency.

In the second step, we select from the computed set of connected subgraphs a minimum order subset with all colors. Clearly, those subgraphs correspond to the connected components of some optimal solution, which can be retrieved by stripping edges with redundant colors. The second step reduces to Minimum-Weight Set Cover when we consider the induced subgraphs as sets (of colors) which are weighted (by the order of the subgraph). We first describe an algorithm for Minimum-Weight Exact Cover using fast subset convolution and then use it to solve Minimum-Weight Set Cover. To improve efficiency, we apply techniques by Björklund et al. [2].

Step one: enumerating induced subgraphs. We make use of the following lemma:

Lemma 1 ([10, Lemma 2]). *Let G be a graph with maximum degree Δ and let v be a vertex in G. There are at most $4^k \cdot (\Delta-1)^k$ connected (induced) subgraphs of G that contain v and have order at most k. Furthermore, these subgraphs can be enumerated in $O(4^k \cdot (\Delta-1)^k \cdot n)$ time.*

Clearly, we can enumerate all connected induced subgraphs of G of order at most k by applying Lemma 1 for each vertex $v \in V(G)$.

Step two: Minimum-Weight Set Cover. We consider Minimum-Weight Set Cover instances with input sets $\mathcal{C} = \{C_1, \ldots, C_m\}$ and weight function w, where n denotes the cardinality of the ground set $U := \bigcup_{C_i \in \mathcal{C}} C_i$, and $w(\mathcal{C}')$ for $\mathcal{C}' \subseteq \mathcal{C}$ denotes the sum of weights of the sets in $\mathcal{C}'$.

Minimum-Weight Set Cover can be solved in $O(2^m)$ time using polynomial space by exhaustive search and in $O(2^n m)$ time using exponential space by dynamic programming. Cygan et al. [4] presented a polynomial-space algorithm

with running time $O^*(\min\{4^n m^{\log n}, 9^n\})$. For our application of Minimum-Weight Set Cover these algorithms are somewhat ill-suited since m may be potentially as large as 2^n, resulting in $4^n \cdot n^{O(1)}$ running times. Better algorithms are known for the unweighted Minimum Set Cover problem which can be solved in $O(2^{0.299(n+m)})$ time [6] and $2^n n^{O(1)}$ time [3], where the second running time avoids the m factor. In the following, we use *fast subset convolution* [2] to obtain an $2^n(nW)^{O(1)}$-time algorithm for Minimum-Weight Set Cover, where W is the maximum weight.

We use the following lemma due to Björklund et al. [2]:

Lemma 2 ([2]). *Consider a set U with $|U| = n$ and a mapping $Q : 2^U \to \{0, \ldots, W\}$. The mapping Q^1 with $Q^1[U'] = \min_{U'' \subseteq U'}(Q[U''] + Q[U' \setminus U''])$ for every $U' \subseteq U$ is called the* convolution *of Q and can be computed in $O(2^n n^3 W \log^2(nW))$ time.*

Björklund et al. [2] did not give precise running time estimates, but Lemma 2 can be derived using their Theorem 1, assuming $O(n \log^2 n)$ time for addition and multiplication of n-bit integers.

As Björklund et al. [2] noted, partitioning problems over the set U can be solved by computing multiple convolutions. We describe in the following the algorithm for Minimum-Weight Exact Cover (the variant of weighted Set Cover where each element needs to be covered by exactly one set) and then how to use the result to solve Minimum-Weight Set Cover.

Minimum-Weight Exact Cover
Instance: A family $\mathcal{C}$ of sets with weight function $w : \mathcal{C} \to \{0, \ldots, W\}$.
Task: Find a minimum-weight subfamily $\mathcal{S} \subseteq \mathcal{C}$ such that each element of $\bigcup_{C_i \in \mathcal{C}} C_i$ occurs in exactly one set in $\mathcal{S}$.

Lemma 3. Minimum-Weight Exact Cover *with weight function $w : \mathcal{C} \to \{0, \ldots, W\}$ can be solved in $O(2^n \cdot n^3 W \log(n) \log^2(nW))$ time.*

Proof. We define an *x-cover* of a subset $U' \subseteq U$ to be a minimum-weight subfamily $\mathcal{C}' \subseteq \mathcal{C}$ containing at most x sets such that each element of U' occurs in exactly one set of $\mathcal{C}'$ and $\bigcup_{C_i \in \mathcal{C}'} C_i = U'$. In these terms Minimum-Weight Exact Cover is to find an n-cover for U.

Consider a mapping $Q : 2^U \to \{0, \ldots, W\}$ and let initially $Q[C_i] = w(C_i)$ for $C_i \in \mathcal{C}$ and $Q[U'] = \infty$ for the remaining $U' \subseteq U$. Now let Q^x denote the mapping resulting from x consecutive convolutions of Q, that is, $Q^0 = Q$ and Q^{x+1} is the convolution of Q^x. We prove by induction that (for all $U' \subseteq U$ and all $x \geq 0$) $Q^x[U']$ is the weight of a 2^x-cover for U' if such a cover exists and $Q^x[U'] = \infty$ otherwise. This implies in particular that $Q^{\lceil \log_2 n \rceil}[U]$ is the weight of an optimal solution to $\mathcal{C}$, if a solution exists.

Clearly the mapping $Q^0 = Q$ meets the claim. Assume that $Q^x[U']$ is the weight of a 2^x-cover for $U' \subseteq U$ if such a cover exists, and $Q^x[U'] = \infty$ otherwise. Now let $\mathcal{C}'$ be a 2^{x+1}-cover for some $U' \subseteq U$. Let $\mathcal{C}_\alpha, \mathcal{C}_\beta \subseteq \mathcal{C}'$ be disjoint subfamilies, $\mathcal{C}_\alpha \cup \mathcal{C}_\beta = \mathcal{C}'$, such that $|\mathcal{C}_\alpha| \leq 2^x$ and $|\mathcal{C}_\beta| \leq 2^x$. (If $|\mathcal{C}'| = 1$, then

$\mathcal{C}_\alpha = \mathcal{C}'$ and $\mathcal{C}_\beta = \emptyset$). Let $U_\alpha = \bigcup_{C_i \in \mathcal{C}_\alpha} C_i, U_\beta = \bigcup_{C_i \in \mathcal{C}_\beta} C_i$. Now $\mathcal{C}_\alpha$ is a 2^x-cover for U_α: it covers each element of U_α exactly once, and if there was an exact cover with lower weight, we could combine it with $\mathcal{C}_\beta$ to get an exact cover for $\bigcup_{C_i \in \mathcal{C}'} C_i$ with lower weight than $\mathcal{C}'$, contradicting that $\mathcal{C}'$ is a 2^{x+1}-cover. The same holds for $\mathcal{C}_\beta$. Hence, $Q^x[U_\alpha] = w(\mathcal{C}_\alpha)$ and $Q^x[U_\beta] = w(\mathcal{C}_\beta)$, therefore $w(\mathcal{C}') = Q[U_\alpha] + Q[U_\beta]$, and due to the minimality of $w(\mathcal{C}')$ we obtain (by convolution) $Q^{x+1}[U'] = \min_{U'' \subseteq U'}(Q[U''] + Q[U' \setminus U'']) = w(\mathcal{C}')$. So $Q^{x+1}[U']$ is the weight of a 2^{x+1}-cover for U'. If no 2^{x+1}-cover for U' exists, then there is no $U'' \subseteq U'$ such that $Q^x[U''] \neq \infty$ and $Q^x[U' \setminus U''] \neq \infty$, hence $Q^{x+1}[U'] = \infty$.

To retrieve the actual solution family, we search for some $U' \subseteq U$ such that $Q^{\lceil \log_2 n \rceil}[U'] + Q^{\lceil \log_2 n \rceil}[U \setminus U'] = Q^{\lceil \log_2 n \rceil}[U]$. We repeat this step for U' and $U \setminus U'$ recursively, until we obtain subsets of U that have a 1-cover. The union of those 1-covers are the sets of the solution family.

The initial mapping Q can be constructed within $O(nm) = O(2^n n)$ time. Next, we compute $\lceil \log_2 n \rceil$ convolutions of Q, each of which takes $O(2^n n^3 W \log^2(nW))$ time, by Lemma 2. Retrieving the solution family takes $O(2^n n)$ time, so we obtain an overall running time of $O(2^n n^3 W \log(n) \log^2(nW))$. □

To convert a table of minimum exact cover weights to a table of minimum (not necessarily exact) cover weights, we iterate over each set $U' \subseteq U$ in increasing order of size, and for each $u \in U'$ replace $Q[U']$ by $\min(Q[U'], Q[U' \setminus \{u\}])$. Together with Lemma 1, this concludes the proof of Theorem 3.

For acyclic inputs, we can use dynamic programming to speed up the enumeration of connected rainbow subgraphs of G, avoiding the dependency on Δ.

Theorem 4. *For an acyclic instance* (G, col) *of* Minimum Rainbow Subgraph *an optimal solution can be computed within* $O(3^p pn + 2^p p^3 n \log^2(pn))$ *time.*

4 Parameterization by Number of Vertex Deletions

In this section, we consider the dual parameter $\ell := n - k$ (where k is the solution order and n is the order of the input graph), that is, the number of vertices that are *not* part of a solution and thus are "deleted" from the input graph. In Section 3, we showed that Rainbow Subgraph is W[1]-hard for the parameter k, but that it becomes fixed-parameter tractable for the parameter (Δ, k). We show that both results also hold when replacing k by ℓ. Hence, parameter ℓ is useful when we ask for the existence of relatively large solutions in sparse graphs.

In contrast to the parameter k, for which Rainbow Subgraph becomes fixed-parameter tractable on trees, we observe W[1]-hardness for parameter ℓ even on very restricted input trees.

Theorem 5. Rainbow Subgraph *is W[1]-hard with respect to the dual parameter* ℓ *even when the input is a tree of height three and every color occurs at most twice.*

By Theorem 5, parameterization by ℓ alone does not yield fixed-parameter tractability. Hence, we consider combinations of ℓ with two parameters. One is

the maximum degree Δ, and the other one is the *maximum color degree* $\Delta_C := \max_{v \in V} |\{c \mid \exists \{u, v\} \in E : \mathrm{col}(\{u, v\}) = c\}|$, which is the maximum number of colors incident with any vertex in G. This parameter was also considered by Schiermeyer [13] for obtaining bounds on the size of minimum rainbow subgraphs. Note that the maximum color degree is upper-bounded by both the maximum degree and by the number of colors in G and that it may be much smaller than either parameter.

First, we show that for the combined parameter (Δ, ℓ) the problem has a polynomial-size problem kernel. To our knowledge, this is the first non-trivial kernelization result for Rainbow Subgraph. As it is common for kernelizations, it is based on a set of data reduction rules. The main idea of the kernelization is as follows. We first remove edges whose colors appear very often compared to Δ and ℓ. Afterwards, deleting any vertex v "influences" only a bounded number of other vertices: at most Δ edges are incident with v and for each of these edges the number of other edges that have the same color depends only on Δ and ℓ. We then consider some vertices that are in every rainbow cover. To this end, we call a vertex v *obligatory* if there is some edge color such that all edges with this color are incident with v. In the data reduction rules, we reduce those obligatory vertices that have only obligatory neighbors. Together with the previous reduction rules, we then obtain the kernel by the following argument: If there are many non-obligatory vertices, then we can find a greedy solution since any vertex deletion has bounded "influence". Otherwise, the overall instance size is bounded as every other vertex is a neighbor of some non-obligatory vertex and each non-obligatory vertex has at most Δ neighbors.

As mentioned above, the first rule removes edges whose color appears very often compared to Δ and ℓ.

Rule 1. *If there is an edge color c such that there are more than $\Delta\ell$ edges with color c, then remove all edges with color c from G.*

We now deal with obligatory vertices. The first simple rule identifies edge colors that are already covered by obligatory vertices.

Rule 2. *If G contains an edge $\{u, v\}$ of color c such that u and v are obligatory, then remove all other edges with color c from G.*

We now work on instances that are reduced with respect to Rule 2. Observe that in such instances every edge between two obligatory vertices has a unique color. This observation is crucial for showing the correctness of the following rules. Their aim is to remove obligatory vertices that have only obligatory neighbors. When removing a vertex in these rules, we decrease k and n by one, thus the value of ℓ remains the same. The correctness of the first rule is obvious.

Rule 3. *Let (G, col) be an instance that is reduced with respect to Rule 2. Then, remove all connected components of G that consist of obligatory vertices only.*

The next two rules remove edges between obligatory vertices.

Rule 4. *Let* (G, col) *be an instance of* Rainbow Subgraph *that is reduced with respect to Rule 2. If* G *contains three obligatory vertices* u*,* v*, and* w *such that* $\{u, v\}, \{v, w\} \in E$ *and* u *has only obligatory neighbors, then remove* $\{u, v\}$ *from* G*. If* u *has degree zero now, then remove* u *from* G*.*

Rule 5. *Let* (G, col) *be an instance of* Rainbow Subgraph *that is reduced with respect to Rule 2. If* G *contains four obligatory vertices* u*,* v*,* w *and* x *such that* $\{u, v\} \in G$ *and* $\{w, x\}$ *in* G *and* u *and* x *have only obligatory neighbors, then do the following.*

Remove $\{w, x\}$ *from* G*. If* v *and* w *are not adjacent, then insert* $\{v, w\}$ *and assign it a unique color. If* x *has now degree zero, then remove* x *from* G*.*

Note that application of Rule 4 does not increase the maximum degree of the instance and decreases the degree of v and w. Furthermore, note that application of Rule 5 may increase the degree of v by one but directly triggers an application of Rule 4 which reduces the degree of v and u again by one. Hence, both rules can be exhaustively applied without increasing the overall maximum degree.

We now show that the instance either has a rainbow cover or that it has bounded size.

Lemma 4. *Let* (G, col) *be an instance that is reduced with respect to Rules 1 to 5. Then,* (G, col) *is a yes-instance or it contains at most* $2\Delta \cdot (\Delta + 1) \cdot \Delta_C \cdot \ell^2$ *vertices.*

Proof. We consider a special type of vertex sets that can be safely deleted. To this end, call a vertex set S a *colorful packing* if

1. no vertex in S is obligatory, and
2. for all u and v in S the set of colors incident with u is disjoint from the set of colors incident with v.

Assume that (G, col) has a *colorful packing* of size ℓ. Then, $G - S$ is a rainbow cover of order k: For each color incident with some vertex v in S, there are two other vertices in V that are connected by an edge with this color (as v is not obligatory). By the second condition, these two vertices are not in S. Hence, this edge color is contained in $G - S$. Summarizing, if (G, col) contains a colorful packing of size at least ℓ, then (G, col) is a yes-instance.

Now, assume that a maximum-cardinality colorful packing S in G has size less than ℓ. Each vertex in S is incident with at most Δ_C colors. For each of these colors, the graph induced by the edges of this color has at most $\Delta\ell$ edges and thus at most $2\Delta\ell$ vertices, since the instance is reduced with respect to Rule 1.

Let T denote the set of vertices in $V \setminus S$ that are incident with at least one edge that has the same color as as an edge incident with some vertex in S. By the above discussion,

$$|T| \leq 2\Delta \cdot \Delta_C \cdot \ell \cdot (\ell - 1).$$

Note that T includes all neighbors of vertices in S. By the maximality of S, all vertices in $V \setminus (S \cup T)$ are obligatory. Now partition $V \setminus (S \cup T)$ into the set X

that has neighbors in T and the set Y that has only neighbors in $(X \cup Y)$. The set X has size at most $(2\Delta_C \cdot \Delta \cdot \ell \cdot (\ell-1)) \cdot \Delta$ since the maximum degree in G is Δ. The set Y has size at most 1 since otherwise one of the Rules 3 to 5 applies (we omit the details).

Hence, since S has size at most $\ell - 1$, G contains at most

$$\ell - 1 + 2\Delta \cdot \Delta_C \cdot \ell \cdot (\ell-1) + 2\Delta^2 \cdot \Delta_C \cdot \ell \cdot (\ell-1) + 1 < 2\Delta \cdot (\Delta+1) \cdot \Delta_C \cdot \ell^2$$

vertices. Thus, if any instance contains more vertices, then a colorful packing of size at least ℓ exists and the instance is a yes-instance. □

Using Lemma 4, we obtain the following theorem.

Theorem 6. Rainbow Subgraph *admits a problem kernel with at most* $2\Delta \cdot (\Delta+1) \cdot \Delta_C \cdot \ell^2$ *vertices that can be computed in* $O(m^2 + mn)$ *time.*

We now consider parameterization by (Δ_C, ℓ) (recall that the color degree Δ_C can be much smaller than Δ). First, by performing the following additional rule, we can use the kernelization result for (Δ, ℓ) to obtain a polynomial problem kernel for (Δ_C, ℓ).

Rule 6. *If G contains a vertex v such that at least $\ell + 2$ edges incident with v have the same color c, then delete an arbitrary one of these edges.*

Rule 6 can be exhaustively performed in linear time. Afterwards, the maximum degree Δ of G is at most $\Delta_C \cdot (\ell+1)$. In combination with Theorem 6, this immediately implies the following.

Theorem 7. Rainbow Subgraph *has a problem kernel with at most* $2(\Delta_C + 1)^3\ell^2(\ell+1)^2$ *vertices that can be computed in* $O(m^2 + mn)$ *time.*

Finally, we describe a simple branching for the parameter (Δ_C, ℓ). Herein, *deleting a vertex* means to remove it from G and to decrease ℓ by one; thus, a deleted vertex is *not* part of a rainbow cover of order k of the original instance.

Branching Rule 1. *If G contains a non-obligatory vertex u, then branch into the following cases. First, recursively solve the instance obtained from deleting u from G. Then, for each color c that is incident with u pick an edge $\{v, w\}$ with color c. If v (w) is non-obligatory, then recursively solve the instance obtained from deleting v (w).*

Note that the parameter ℓ decreases by one in each branch. Exhaustively applying Branching Rule 1 until either every vertex is obligatory or $\ell \leq 0$ yields an algorithm with the following running time.

Theorem 8. Rainbow Subgraph *can be solved in* $O((2\Delta_C + 1)^\ell \cdot (n+m))$ *time.*

5 Outlook

Considering its biological motivation, it would be interesting to gain further, potentially data-driven parameterizations of Minimum Rainbow Subgraph that may help identifying further practically relevant and tractable special cases. From a more graph-theoretic point of view, we left open a deeper study of parameters measuring the degree of acyclicity of the underlying graph, such as treewidth or feedback set numbers. A further question is whether for our fixed-parameter tractability result in Theorem 3 we can avoid exponential memory consumption.

Acknowledgments. We thank the reviewers of WG' 14 for their thorough and valuable feedback.

References

[1] P. Alimonti and V. Kann. Some APX-completeness results for cubic graphs. *Theor. Comput. Sci.*, 237(1-2):123–134, 2000.
[2] A. Björklund, T. Husfeldt, P. Kaski, and M. Koivisto. Fourier meets Möbius: Fast subset convolution. In *Proc. 39th STOC '07*, pages 67–74. ACM, 2007.
[3] A. Björklund, T. Husfeldt, and M. Koivisto. Set partitioning via inclusion-exclusion. *SIAM J. Comput.*, 39(2):546–563, 2009.
[4] M. Cygan, Ł. Kowalik, and M. Wykurz. Exponential-time approximation of weighted set cover. *Inform. Process. Lett.*, 109(16):957–961, 2009.
[5] R. Fernandes and S. Skiena. Microarray synthesis through multiple-use PCR primer design. *Bioinformatics*, 18(suppl 1):128–135, 2002.
[6] F. V. Fomin, F. Grandoni, and D. Kratsch. A measure & conquer approach for the analysis of exact algorithms. *J. ACM*, 56(5):25:1–25:32, 2009.
[7] M. T. Hajiaghayi, K. Jain, L. C. Lau, I. I. Mandoiu, A. Russell, and V. V. Vazirani. Minimum multicolored subgraph problem in multiplex PCR primer set selection and population haplotyping. In *Proc. 6th ICCS '06*, volume 3992 of *LNCS*, pages 758–766. Springer, 2006.
[8] J. Katrenič and I. Schiermeyer. Improved approximation bounds for the minimum rainbow subgraph problem. *Inform. Process. Lett.*, 111(3):110–114, 2011.
[9] M. Koch, S. Matos Camacho, and I. Schiermeyer. Algorithmic approaches for the minimum rainbow subgraph problem. *ENDM*, 38:765–770, 2011.
[10] C. Komusiewicz and M. Sorge. Finding dense subgraphs of sparse graphs. In *Proc. 7th IPEC*, volume 7535 of *LNCS*, pages 242–251. Springer, 2012.
[11] S. Matos Camacho, I. Schiermeyer, and Z. Tuza. Approximation algorithms for the minimum rainbow subgraph problem. *Discrete Math.*, 310(20): 2666–2670, 2010.
[12] A. Popa. Better lower and upper bounds for the minimum rainbow subgraph problem. *Theor. Comput. Sci.*, 543:1–8, 2014.
[13] I. Schiermeyer. On the minimum rainbow subgraph number of a graph. *Ars Mathematica Contemporanea*, 6(1), 2012.
[14] D. P. Williamson and D. B. Shmoys. *The Design of Approximation Algorithms.* Cambridge University Press, 2011.