Graph Motif Problems Parameterized by Dual[1]

Guillaume Fertin[1] *Christian Komusiewicz*[2]

[1]Laboratoire des Sciences du Numérique de Nantes, UMR CNRS 6004,
Université de Nantes, 2 rue de la Houssinière, 44322 Nantes Cedex 3, France
[2]Fachbereich Mathematik und Informatik, Philipps-Universität Marburg,
Germany

Abstract

Let $G = (V, E)$ be a vertex-colored graph, where C is the set of colors used to color V. The GRAPH MOTIF (or GM) problem takes as input G, a multiset M of colors built from C, and asks whether there is a subset $S \subseteq V$ such that (i) $G[S]$ is connected and (ii) the multiset of colors obtained from S equals M. The COLORFUL GRAPH MOTIF (or CGM) problem is the special case of GM in which M is a set, and the LIST-COLORED GRAPH MOTIF (or LGM) problem is the extension of GM in which each vertex v of V may choose its color from a list $\mathcal{L}(v) \subseteq C$ of colors.

We study the three problems GM, CGM, and LGM, parameterized by the dual parameter $\ell := |V| - |M|$. For general graphs, we show that, assuming the strong exponential time hypothesis, CGM has no $(2 - \epsilon)^\ell \cdot |V|^{\mathcal{O}(1)}$-time algorithm, which implies that a previous algorithm, running in $\mathcal{O}(2^\ell \cdot |E|)$ time is optimal [Betzler et al., IEEE/ACM TCBB 2011]. We also prove that LGM is W[1]-hard with respect to ℓ even if we restrict ourselves to lists of at most two colors. Finally, we show that if the input graph is a tree, then GM can be solved in $\mathcal{O}(3^\ell \cdot |V|)$ time but admits no polynomial-size problem kernel, while CGM can be solved in $\mathcal{O}(\sqrt{2}^\ell + |V|)$ time and admits a polynomial-size problem kernel.

[1]A preliminary version of this work appeared in *Proceedings of the 27th Annual Symposium on Combinatorial Pattern Matching (CPM '16)*, volume 54 of LIPIcs, pages 7:1–7:12. This version contains all missing proofs, an improved running time for Theorem 4 and a new tractability result (Theorem 5).

1   Introduction

The SUBGRAPH ISOMORPHISM problem is the following pattern matching problem in graphs: given a (typically large) host graph G and a (small) query graph H, return one (or all) occurrence(s) of H in G, where the term occurrence denotes here a subset S of $V(G)$ such that $G[S]$, the subgraph of G induced by S, is isomorphic to H. This type of graph mining problem has different applications, notably in biology [26]. SUBGRAPH ISOMORPHISM is a *structural* graph pattern matching problem, where one looks for similar graph structures between H and G. In some biological contexts, however, additional information is provided to the vertices of the graphs, for example their biological function. This can be modeled by labeling each vertex of the graph, for example by giving it one or several colors, each corresponding to an identified function. In the presence of such functional annotation, the structure of a given induced subgraph may be of less importance than the functions it corresponds to. Thus, a new set of *functional* graph pattern matching problems has emerged, starting with the GRAPH MOTIF problem [21], which was introduced in the context of the analysis of metabolic networks. In GRAPH MOTIF, the query is a multiset M of colors that represents the functions of interest, and we search for an occurrence of M in the host graph, where the previous demand of being isomorphic to the query is replaced by a connectivity demand.

> GRAPH MOTIF (GM)
> **Input:** A multiset M built on a set C of colors, an undirected graph $G = (V, E)$, and a coloring $\chi : V \to C$.
> **Question:** Is there a set $S \subseteq V$ such that $G[S]$ is connected and there is a one-to-one mapping $f : S \to M$ such that $f(v) = \chi(v)$ for all $v \in S$?

Throughout this work, we will assume that each color of C appears at least once in M—otherwise, every color in C that is not in M, together with the vertices of G that are assigned this color, may be safely removed from the instance. Many variants of GM have been introduced and studied. In particular, LIST-COLORED GRAPH MOTIF (or LGM) is a generalization of GM that is used to identify, in a given protein interaction network, protein complexes that are similar to a given protein complex from a different species [7]. In LGM, the graph G is associated with a list-coloring $\mathcal{L} : V \to 2^C$, that is, each vertex v is associated with a set $\mathcal{L}(v)$ of colors, and the question is whether there is a set $S \subseteq V$ such that (i) $G[S]$ is connected and (ii) the one-to-one mapping f from S to M we look for satisfies $\forall v \in S : f(v) \in \mathcal{L}(v)$. The special case of GM in which M is a set is called COLORFUL GRAPH MOTIF (or CGM). Many optimization problems related to GM have received interest, including some that are related to metabolite identification through tandem mass spectrometry and where the input graph is directed and edge-weighted [25]. All these problem variants have given rise to an abundant literature. CGM, GM, and LGM are NP-hard even in very restricted cases [21, 12, 6]. Consequently, many of the above-mentioned studies have focused on (dis)proving fixed-parameter

tractability of the problems (see e.g. [27] for an informal survey on the topic). In such cases, very often the parameter $k := |M| = |S|$ is considered.

In this paper, we study the parameterized complexity of GM, CGM, and LGM, but our approach differs from the usual viewpoint by focusing on the *dual* parameter $\ell := |V| - |S|$, that is, ℓ is the number of vertices to be deleted from G to obtain a solution. Although the choice of ℓ may be disputable because *a priori* it may be too large to expect a good behavior in practice, there are several arguments for choosing such a parameter: First, after some initial data reduction, the input may be divided into smaller connected components, where ℓ is not much larger than k. Second, the algorithms for parameter k rely on algebraic techniques or dynamic programming, and in both cases, the worst-case running time is equivalent to the actual running time. In contrast, for example for CGM, the algorithm for parameter ℓ is a search tree algorithm [2], and search tree algorithms can be substantially accelerated via pruning rules. Finally, there are subgraph mining problems where the dual parameter ℓ is usually bigger than the parameter k but nevertheless leads to the current-best algorithms (in terms of performance on real-world instances), see e.g. [18, 20]. Hence, parameterization by ℓ may be useful even if ℓ is bigger than k, and thus deserves to be studied.

Related work and our contribution. GM is NP-hard, even when M is composed of two colors [12]. Concerning the parameterized complexity for parameter $k := |M|$, the current-best randomized algorithm has a running time of $2^k \cdot n^{\mathcal{O}(1)}$ [3, 24] where $n := |V|$, and there is evidence that this cannot be improved to a running time of $(2 - \epsilon)^k \cdot n^{\mathcal{O}(1)}$ [3]. The current-best running time for a deterministic algorithm is $5.18^k \cdot n^{\mathcal{O}(1)}$ [23]. GM on trees can be solved in $n^{\mathcal{O}(c)}$ time where c is the number of colors in M [12], but is W[1]-hard with respect to c [12]. Other parameters, essentially related to the structure of the input graph G, have been studied by Ganian [17], Bonnet and Sikora [6], and Das et al. [9]. For example, GRAPH MOTIF is fixed-parameter tractable when parameterized by the size of a vertex cover of the input graph [17, 6]. Finally, concerning parameter ℓ, GM has been shown to be W[1]-hard, even when M is composed of two colors [2].

Since CGM is a special case of GM, any above-mentioned positive result for GM also holds for CGM. In addition, CGM is NP-hard even for trees of maximum degree 3 [12], and does not admit a polynomial-size problem kernel with respect to k even if G has diameter two or if G is a comb graph (a special type of tree with maximum degree 3) [1]. Finally, CGM can be solved in $\mathcal{O}(2^\ell \cdot m)$ time [2], where $m := |E|$. The LGM problem is an extension of GM and thus any negative result for GM propagates to LGM. Moreover, LGM is fixed-parameter tractable with respect to k, and the current-best algorithm runs in $2^k \cdot n^{\mathcal{O}(1)}$ time [24]. Concerning parameter ℓ, LGM has been shown to be W[1]-hard even when M is a set [2].

As mentioned above, we study GM, LGM, and CGM with respect to the dual parameter $\ell := n - k$. Since many results in general graphs turn out to be negative, we also study the special case where the input graph G is a tree.

Table 1: Overview of new and previous results with respect to the dual parameter $\ell := n - k$, where $n := |V|$, $k := |M|$, $m := |E|$ and $\Delta := \max_{v \in V} |\mathcal{L}(v)|$ denotes the maximum list size in G. The lower bound result for CGM assumes the strong exponential time hypothesis (SETH) [19], the results showing non-existence of polynomial kernels assume that NP $\subsetneq$ coNP/poly.

	General graphs	Trees
LGM	W[1]-hard [2]	?
LGM, $\Delta = 2$	W[1]-hard (Cor. 1)	?
GM	W[1]-hard [2]	$\mathcal{O}(3^\ell \cdot n)$ (Thm. 4)
		no poly. kernel (Thm. 6)
CGM	$\mathcal{O}(2^\ell \cdot m)$ [2],	$\mathcal{O}(\sqrt{2}^\ell + n)$ (Thm. 8),
	no $(2 - \epsilon)^\ell \cdot n^{\mathcal{O}(1)}$ algo. (Thm. 1)	
	no poly. kernel (Thm. 2)	$(2\ell + 1)$-vert. kernel (Thm. 7)

Our results are summarized in Table 1. In a nutshell, we strengthen previous hardness results for the general case and show that the $\mathcal{O}(2^\ell \cdot m)$-time algorithm for CGM is essentially optimal. Then, we show that for GM on trees and for some special cases of LGM on trees, a fixed-parameter algorithm can be achieved. Finally, we show that for CGM on trees, a polynomial-size problem kernel and better running times than for general graphs can be achieved.

Preliminaries.   For an integer n, we use $[n] := \{1, \ldots, n\}$ to denote the set of the integers from 1 through n. Throughout the paper, the input graph for our three problems is $G = (V, E)$, and we let $n := |V|$ and $m := |E|$ denote its number of vertices and edges, respectively. For a vertex set $S \subseteq V$, we let $G[S] := (S, \{\{u, v\} \in E \mid u, v \in S\})$ denote the subgraph of G induced by S. The set S of vertices sought for in the three problems is called an *occurrence* of M. If G is vertex-colored, we call a vertex set S *colorful* if $|S| = |M|$ and all vertices in S have pairwise different colors. A vertex v is called *unique* if it is assigned a color c that is assigned to no other vertex in V; the color c is also called unique in this case. For a multiset M and an element c of M, we use $M(c)$ to denote the multiplicity of c in M.

To analyze the structure of the coloring constraints for instances of LGM, we consider the following auxiliary graph.

Definition 1 *Let $(M, G, \mathcal{L})$ be an instance of* LGM. *The vertex-color graph H of $(M, G, \mathcal{L})$ is the bipartite graph with vertex set $V \cup C$ and edge set $\{\{v, c\} \mid v \in V, c \in \mathcal{L}(v)\}$.*

Observe that GM instances are LGM instances where each vertex from V has degree one in the vertex-color graph H. In other words, H is a disjoint union of stars whose non-leaves are vertices from C. Moreover, an LGM instance

where H is a disjoint union of bicliques can be easily replaced by an equivalent GM instance: For each biclique K in H, replace the color set $K \cap C$ by one color with multiplicity $\sum_{c \in K \cap C} M(c)$ in M and assign this color to all vertices in $K \cap V$.

We briefly recall the relevant notions of parameterized algorithmics [8, 11]. Parameterized algorithmics aims at analyzing the impact of structural input properties on the difficulty of computational problems. Formally, a parameterized problem L is a subset of $\Sigma^* \times \mathbb{N}$ where the second component is the *parameter*. A parameterized problem L is fixed-parameter tractable if every input instance (I, k) can be solved in $f(k) \cdot |I|^{\mathcal{O}(1)}$ time where f is a computable function depending only on k. The class of fixed-parameter tractable problems is called FPT.

A *reduction to a problem kernel*, or *kernelization*, is an algorithm that takes as input an instance (I, k) of a parameterized problem and produces in polynomial time an instance (I', k') such that

- (I, k) is a yes-instance if and only if (I', k') is a yes-instance and

- $|I'| + k' \leq g(k)$ where g is a computable function depending only on k.

The instance (I', k') is called *problem kernel* and g is called the *size of the problem kernel*. If g is a polynomial function, then the problem admits a *polynomial-size problem kernelization*.

The class W[1] is a class of parameterized problems that contains all fixed-parameter tractable problems, that is, FPT $\subseteq$ W[1]. The basic assumption of parameterized complexity theory is FPT $\neq$ W[1] [8, 11]. Consequently, if a parameterized problem is W[1]-hard, then we assume that it cannot be solved in $f(k) \cdot n^{\mathcal{O}(1)}$ time [8, 11]. The strong exponential time hypothesis (SETH) assumes that, for any $\epsilon > 0$, CNF-SAT cannot be solved in time $(2 - \epsilon)^n \cdot |\Phi|^{\mathcal{O}(1)}$ where Φ is the input formula and n is the number of variables [19].

This work is structured as follows. In Section 2, we present lower bounds for LGM and CGM on general graphs. These negative results motivate our study of the case when G is a tree; our results for GM on trees and CGM on trees will be presented in Section 3 and Section 4, respectively. We conclude with an outlook of future work in Section 5.

2 Parameterization by Dual in General Graphs: Tight Lower Bounds

CGM can be solved in $\mathcal{O}(2^\ell \cdot m)$ time [2]. We show that this running time bound is essentially optimal.

Theorem 1 COLORFUL GRAPH MOTIF *cannot be solved in $(2 - \epsilon)^\ell \cdot n^{\mathcal{O}(1)}$ time unless the strong exponential time hypothesis (SETH) fails.*

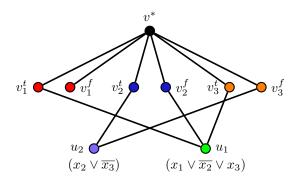**Proof:** We present a polynomial-time reduction from CNF-SAT:

Figure 1: The construction for the input formula $\Phi = (x_2 \vee \overline{x_3}) \wedge (x_1 \vee \overline{x_2} \vee x_3)$.

Input: A boolean formula Φ in conjunctive normal form with clauses $\mathcal{C}_1, \ldots, \mathcal{C}_q$ over variable set $X = \{x_1, \ldots, x_r\}$.
Question: Is there an assignment $\beta : X \rightarrow \{\texttt{true}, \texttt{false}\}$ that satisfies Φ?

The reduction works as follows; for an example see Figure 1. First, for each variable $x_i \in X$ introduce two *variable vertices* v_i^t and v_i^f and color each of them with color χ_i^X. The idea is that (with the final occurrence) we must select exactly one vertex for this color. This selection will correspond to a truth assignment to X. Now, introduce for each clause $\mathcal{C}_i$ a *clause vertex* u_i, color u_i with a unique color $\chi_i^\mathcal{C}$ and make u_i adjacent to vertex v_j^t if x_j occurs non-negated in $\mathcal{C}_i$ and to vertex v_j^f if x_j occurs negated in $\mathcal{C}_i$. Finally, introduce one further vertex v^* with a unique color χ^*, make v^* adjacent to all variable vertices and let M be the set containing each of the introduced colors exactly once. Note that there are exactly $|X|$ colors that appear twice in G and that all other colors appear exactly once. Hence, $\ell = |X|$. We next show the correctness of the reduction. Let $I = (M, G, \chi)$ denote the constructed instance of CGM.

First, assume that Φ is satisfiable and let β be a satisfying assignment of X. For the CGM instance I consider the vertex set $S \subseteq V$ that contains all clause vertices, vertex v^*, and for each variable x_i the vertex v_i^t if β sets x_i to $\texttt{true}$ and the vertex v_i^f otherwise. Clearly, $|S| = |M|$ and no two vertices of S have the same color. To show that I is a yes-instance of CGM it remains to show that $G[S]$ is connected. First, the subgraph induced by the variable vertices in S plus v^* is a star and thus it is connected. Second, since β is a satisfying assignment each clause vertex in S has at least one neighbor in S (which is by construction a variable vertex). Hence, $G[S]$ is connected.

Conversely, assume that I is a yes-instance of CGM, and let S be a colorful vertex set with $|S| = |M|$ such that $G[S]$ is connected. Since S is colorful, the variable vertices in S correspond to a truth assignment β of X. This assignment satisfies Φ: Indeed, since $G[S]$ is connected, there is a path in $G[S]$ between each clause vertex u_i and v^*, and thus S contains some neighbor of u_i. If this neighbor is v_j^t, then by construction β assigns $\texttt{true}$ to x_j. Otherwise, the neighbor is v_j^f

and β assigns `false` to x_j. In both cases, $\mathcal{C}_i$ is satisfied.

Thus, the two instances are equivalent. Now observe that since $\ell = |X| = r$ and $n = 2r+q+1$, any $(2-\epsilon)^\ell \cdot n^{\mathcal{O}(1)}$-time algorithm implies a $(2-\epsilon)^r \cdot (r+q)^{\mathcal{O}(1)}$-time algorithm for CNF-SAT. This directly contradicts the SETH. □

The above reduction also makes the existence of a polynomial-size problem kernel for parameter ℓ unlikely. This is implied by the following two facts. First, CNF-SAT parameterized by the number of variables does not admit a polynomial-size problem kernel unless NP $\subseteq$ coNP/poly [10]. Second, the reduction presented in the proof of Theorem 1 is a *polynomial parameter transformation* [5] from CNF-SAT parameterized by the number of variables to CGM parameterized by ℓ. More precisely, given an input CNF-SAT formula Φ on variable set X, the reduction produces an instance $I = (M, G, \chi)$ of CGM with $\ell = |X|$. Now, any polynomial-size problem kernelization applied to I produces in polynomial time an equivalent CGM instance I' of size $\ell^{\mathcal{O}(1)} = |X|^{\mathcal{O}(1)}$. Since CNF-SAT is NP-hard, we can now transform this CGM instance in polynomial time into an equivalent CNF-SAT instance that has size $\ell^{\mathcal{O}(1)} = |X|^{\mathcal{O}(1)}$. Hence, a polynomial-size problem kernel for CGM parameterized by ℓ implies a polynomial-size problem kernel for CNF-SAT parameterized by $|X|$. This implies NP $\subseteq$ coNP/poly [10] (which in turn implies a collapse of the polynomial hierarchy).

Theorem 2 COLORFUL GRAPH MOTIF *parameterized by ℓ does not admit a polynomial-size problem kernel unless* NP $\subseteq$ coNP/poly.

We have thus resolved the parameterized complexity of CGM parameterized by ℓ on general graphs and now turn to the more general LGM problem, which is W[1]-hard with respect to ℓ [2]. Here, it would be desirable to obtain fixed-parameter algorithms for parameter ℓ at least for some restricted inputs. In other words, we would like to further exploit the structure of real-world instances to obtain tractability results. A very natural approach is to consider the size and structure of the list-colorings $\mathcal{L}(v)$ as additional parameter. Unfortunately, the problem remains W[1]-hard even for the following very restricted case of list-colorings. Recall that the vertex-color graph is the bipartite graph with vertex set $V \cup C$ in which $v \in V$ and $c \in C$ are adjacent if and only if $c \in \mathcal{L}(v)$.

Theorem 3 LIST-COLORED GRAPH MOTIF *is W[1]-hard with respect to ℓ even if the vertex-color graph is a disjoint union of paths.*

Proof: We reduce from the MULTICOLORED INDEPENDENT SET problem:

> **Input:** An undirected graph $H = (W, F)$ and a vertex-labeling $\lambda : W \to \{1, \ldots, k\}$.
> **Question:** Is there a vertex set $S \subseteq W$ such that $|S| = k$, the vertices in S have pairwise different labels, and $H[S]$ has no edges?

MULTICOLORED INDEPENDENT SET has been shown to be W[1]-hard when parameterized by k [13]. We refer to the colors of the MULTICOLORED INDEPENDENT SET instance as labels to avoid confusion with the colors of the
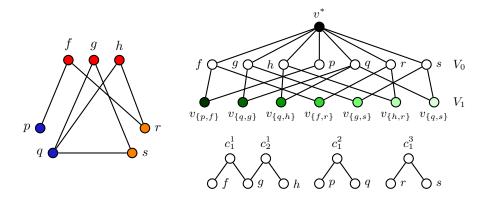
Figure 2: An example of the construction for the reduction from MULTICOLORED INDEPENDENT SET. Left: An instance of MULTICOLORED INDEPENDENT SET where f, g, and h have label 1, p and q have label 2, and r and s have label 3. Right: The constructed graph instance. The graph G with the coloring for those vertices that receive their own private color is shown on top, the vertex-color graph for vertices from V_0 is shown below.

LGM instance. Assume without loss of generality that each label class in H contains the same number x of vertices (this can be achieved by padding smaller classes with additional vertices) and that there is an arbitrary but fixed ordering of the vertices of H.

The reduction works as follows; see Figure 2 for an example. We first describe the input graph G of the LGM instance. We let $V = V_0 \cup V_1 \cup \{v^*\}$, where $V_0 = W$ and $V_1 = \{v_e | e \in F\}$. Now construct the edge set E of G as follows. First, make vertex v^* adjacent to all vertices of V_0. Then, for each edge $\{u, w\}$ of H make vertex $v_{\{u,w\}}$ adjacent to u and w. This completes the construction of G. Now let us describe the coloring of the vertices. We start with the colors given to $V_0 = W$. For each label i from λ do the following: create $x - 1$ colors $c_1^i, \ldots, c_{x-1}^i$. Now, with respect to the above-mentioned ordering, color the first vertex of label class i with color c_1^i, color any jth vertex, $2 \leq j \leq x - 1$, with the list $\{c_{j-1}^i, c_j^i\}$, and finally color the xth vertex with color c_{x-1}^i. Next, color each vertex from $V_1 \cup \{v^*\}$ with a unique color. Let $\mathcal{L}$ denote the list-coloring of $V(G)$ that we just described. We define the motif M as the set containing each color present in $\mathcal{L}$. Clearly, the reduction works in polynomial time. Note that $|V| = kx + |E| + 1$ and $|M| = k(x - 1) + |E| + 1$ and thus $\ell = |V| - |M| = k$. To prove our claim, it thus remains to show the correctness of the reduction:

> (H, k) is a yes-instance of MULTICOLORED INDEPENDENT SET $\Leftrightarrow$
> $(M, G, \mathcal{L})$ is a yes-instance of LGM.

($\Rightarrow$) Let S be a size-k independent set with pairwise different vertex labels in H. Consider the set $Y := V \setminus S$ in G. First, note that $G[Y]$ is connected:

vertex v^* is adjacent to all vertices in $Y \cap V_0$ and each vertex $v_{\{u,w\}}$ of $Y \cap V_1 = V_1$ has at least one neighbor in $Y \cap V_0$, because at most one of the endpoints of $\{u, w\}$ is in the independent set S.

It remains to show that we can assign colors to the vertices such that the union of the vertex colors is M. All vertices with unique colors are contained in Y and their coloring is clear. All other vertices are in V_0. Now consider label class i of V_0. Exactly one vertex u of label class i is contained in S. Let j be the number such that u is the jth vertex of the label class i. Then, color the qth vertex of label class i with color c_q^i if $q < j$ and with color c_{q-1}^i if $q > j$. This coloring assigns $x - 1$ different colors to the vertices of each label class. Hence, there is a coloring of the vertices of Y that is equal to M.

($\Leftarrow$) Let Y denote an occurrence of M in G. First, observe that there are only $x - 1$ colors for the x vertices of each label class. Hence, Y contains exactly $x - 1$ vertices of each label class. Now, let S denote the set containing, for each label class, the only vertex *not* contained in Y. Clearly, $|S| = k$ and the elements of S have pairwise different labels in H. Furthermore, S is an independent set in H: since $G[Y]$ is connected, for each edge vertex $v_{\{u,w\}}$ at least one of its neighbors is in Y. Hence, at most one of the endpoints of each edge $\{u, w\}$ is in S. $\square$

We immediately obtain the following.

Corollary 1 List-Colored Graph Motif *is W[1]-hard with respect to ℓ even if $|\mathcal{L}(v)| \leq 2$ for every vertex v in G.*

3   Graph Motif on Trees

Motivated by the negative results on general graphs from Section 2, we now study the special case where the input graph is a tree. For LGM, we were not able to resolve the parameterized complexity with respect to ℓ for this case. Hence, we focus on the more restricted GM problem. We show that GM is fixed-parameter tractable with respect to ℓ if the input graph is a tree. Recall that for general graphs, GM is W[1]-hard for ℓ even if the motif M contains only two colors [2]. Hence, our result shows that the tree structure significantly helps when parameterizing by ℓ. We then show that the fixed-parameter algorithm for GM on trees extends to some special cases of LGM in which the vertex-color graph is also a tree. Finally, we show that a polynomial-size kernel for GM on trees parameterized by ℓ is unlikely.

3.1   A Dynamic Programming Algorithm

Call a color of M *abundant* if it occurs more often in G than in M. In other words, a color is abundant precisely if we have to delete at least one vertex with this color to obtain a solution S. Let $c_1, \ldots, c_j$ denote the abundant colors of M, and let ℓ_i denote the difference between the number of vertices in V that

have color c_i and the multiplicity $M(c_i)$ of c_i in M. This implies in particular that $\sum_{1 \leq i \leq j} \ell_i = \ell$.

The algorithm is a dynamic programming algorithm that works on a rooted representation T of G. We obtain T by choosing an arbitrary vertex $r \in V$ and rooting G at r. As usual for dynamic programming on trees, the idea is to combine partial solutions of subtrees. Our algorithm is somewhat similar to a previous dynamic programming algorithm for GM on graphs of bounded treewidth [12] but the analysis and concrete table setup is different.

In the following, let T_v denote the subtree of T rooted at vertex v. For each subtree, we let $\text{occ}(T_v, c)$ denote the number of vertices in T_v that have color c. If a solution contains vertices from T_v and further vertices, then it must contain v and all vertices with non-abundant colors in T_v. Hence, in the dynamic programming it is sufficient to consider subtrees described in the following definition.

Definition 2 *We call a connected subtree T' of T_v safe if T' contains v and if every vertex of T_v that is colored by a non-abundant color is contained in T'.*

We fill a family of dynamic programming tables D_v, one table for each $v \in V$. The entries of D_v are defined as follows:

$$D_v[\lambda_1, \ldots, \lambda_j] = \begin{cases} 1 & \text{if } T_v \text{ has a safe subtree containing for each } c_i, \, 1 \leq i \leq j, \\ & \text{exactly } \text{occ}(T_v, c_i) - \lambda_i \text{ vertices of color } c_i, \\ 0 & \text{otherwise.} \end{cases}$$

Assume for now that the table has completely been filled out. Then, it can be directly determined whether G has an occurrence S of M: If S is an occurrence of M, then let v denote the root of $T[S]$. Clearly, $T[S]$ is a safe subtree of T_v. Moreover, every vertex with a non-abundant color is contained in $T[S]$ and for all vertices with an abundant color c_i, the tree $T[S]$ contains $\text{occ}(T_v, c_i) - \lambda_i$ vertices with color c_i for some $\lambda_i \geq 0$. Thus, there is some table entry $D_v[\lambda_1, \ldots, \lambda_j]$ whose value is 1 and where $\text{occ}(T_v, c_i) - \lambda_i$ is the multiplicity of c_i in M for each c_i. Conversely, if there is some entry $D_v[\lambda_1, \ldots, \lambda_j]$ with value 1 such that T_v contains all vertices with non-abundant colors and for each c_i, $1 \leq i \leq j$, $\text{occ}(T_v, c_i) - \lambda_i$ is exactly the multiplicity of c_i in M, then there is at least one safe subtree of T_v whose vertex set is an occurrence of M.

By the above, one may solve GM by filling table D, and then checking for each vertex v whether one of the entries of $D_v[\lambda_1, \ldots, \lambda_j]$ with value 1 implies the existence of a solution. For the running time bound, the main observation that we exploit is that if a safe rooted subtree of T_v contains all the vertices of T_v that are in a solution S, then it contains at least $\text{occ}(T_v, c_i) - \ell_i$ vertices with color c_i. Consequently, the relevant range of values for λ_i is in $[0, \ell_i]$ and thus bounded in the parameter value ℓ.

We now describe how to fill in table D. To initialize D, consider each leaf v of the tree T. By the definition of D, an entry can have value 1 only if there is a corresponding safe tree which needs to contain v. Thus,

$$D_v[\lambda_1, \ldots, \lambda_j] = 1 \Leftrightarrow \lambda_1 = \ldots = \lambda_j = 0.$$

Now, to compute the entries of D for a non-leaf vertex v, we combine the entries of the children of v. To this end, fix an arbitrary ordering of the children of v and denote them by $u_1, \ldots, u_{\deg(v)}$. Now, let T_v^i denote the subtree rooted at v containing the vertices of each T_{u_q}, $1 \leq q \leq i$, and no vertices from each T_{u_q}, $q > i$. For increasing i, we compute solutions for T_v^i, eventually computing the solutions for $T_v^{\deg(v)} = T_v$. To compute these solutions, we define an auxiliary table D_v^i. The table entries are defined just as for D, that is,

$$D_v^i[\lambda_1, \ldots, \lambda_j] = \begin{cases} 1 & \text{if } T_v^i \text{ has a safe subtree containing for each } c_i, \ 1 \leq i \leq j, \\ & \text{exactly } \mathrm{occ}(T_v, c_i) - \lambda_i \text{ vertices of color } c_i, \\ 0 & \text{otherwise.} \end{cases}$$

Observe that, since $T_v^{\deg(v)} = T_v$, we have $D_v^{\deg(v)} = D_v$ and thus by computing $D_v^{\deg(v)}$ we also compute D_v. Now, D_v^1 can be computed in a straightforward fashion from the entries of D_{u_1}.

$$D_v^1[\lambda_1, \ldots, \lambda_j] = \begin{cases} 1 & \text{if } D_{u_1}[\lambda_1, \ldots, \lambda_j] = 1 \\ 1 & \text{if } \mathrm{occ}(T_{u_1}, c_i) = \lambda_i \text{ for every } i, \ 1 \leq i \leq q, \\ & \text{and } T_{u_1} \text{ contains only abundant colors,} \\ 0 & \text{otherwise.} \end{cases}$$

The first case corresponds to the case that the safe subtree T' of T_v^i contains at least one vertex of T_{u_1}, the second case corresponds to the case that T' contains only v.

To compute D_v^i for $i > 1$, we combine entries of D_v^{i-1} with D_{u_i}.

$$D_v^i[\lambda_1, \ldots, \lambda_j] = \begin{cases} 1 & \text{if } T_{u_i} \text{ contains only abundant colors and} \\ & D_v^{i-1}[\lambda_1 - \mathrm{occ}(T_{u_i}, c_1), \ldots, \lambda_j - \mathrm{occ}(T_{u_i}, c_j)] = 1, \\ 1 & \text{if there is } (\lambda_1', \ldots, \lambda_j') \text{ such that} \\ & D_v^{i-1}[\lambda_1', \ldots, \lambda_j'] = D_{u_i}[\lambda_1 - \lambda_1', \ldots, \lambda_j - \lambda_j'] = 1, \\ 0 & \text{otherwise.} \end{cases}$$

The first case corresponds to the situation in which no vertex of T_{u_i} is part of the safe subtree, in the second case the safe subtree contains some vertices of T_{u_i} and some vertices of T_v^{i-1}.

This completes the description of the dynamic programming recurrences. The correctness follows from the fact that the recurrence considers all possible cases to "distribute" the vertex deletions. It remains to bound the running time.

Theorem 4 GRAPH MOTIF *can be solved in* $\mathcal{O}(3^\ell \cdot n)$ *time if G is a tree.*

Proof: As described above, the relevant values of each λ_i are in $[0, \ell_i]$. Thus, for each subtable D_v^i and D_v, the number of entries to compute is $\prod_{1 \leq i \leq j}(\ell_i + 1)$. The dominating term in the overall running time is the computation of the second term in the recurrence for $D_v^i[\lambda_1, \ldots, \lambda_j]$ where we consider all $(\lambda_1', \ldots, \lambda_j')$ such

that $D_v^{i-1}[\lambda_1', \ldots, \lambda_j'] = D_{u_i}[\lambda_1 - \lambda_1', \ldots, \lambda_j - \lambda_j'] = 1$. The number of possible choices can be computed as follows. For each λ_i, the values of λ_i' range between 0 and λ_i. Overall, this gives

$$\sum_{j=0}^{\ell_i} j + 1 = \sum_{j=1}^{\ell_i+1} j = (\ell_i + 2) \cdot (\ell_i + 1)/2$$

possibilities for choosing λ_i and λ_i'. We now bound this product in ℓ. Thus, we aim to find the vector $(\ell_1, \ldots, \ell_j)$ that maximizes $\prod_{1 \leq i \leq j} (\ell_i + 2) \cdot (\ell_i + 1)/2$ under the constraint $\sum_{1 \leq i \leq j} \ell_i = \ell$. We claim that this is the vector with $\ell_1 = \ldots = \ell_j = 1$.

To this end, consider a vector $V = (\ell_1, \ldots, \ell_j)$ whose maximum entry is at least 2. Without loss of generality, assume thus $\ell_j > 1$. Now consider $V' = (\ell_1, \ldots, \ell_j - 1, 1)$ and observe that $(\sum_{1 \leq i < j} \ell_i) + (\ell_j - 1) + 1 = \ell$, that is, V' also satisfies the summation constraint. Moreover,

$$\frac{\prod_{1 \leq i \leq j}((\ell_i + 2) \cdot (\ell_i + 1)/2)}{(\prod_{1 \leq i < j}((\ell_i + 2) \cdot (\ell_i + 1)/2) \cdot ((\ell_j + 1)(\ell_j)/2) \cdot 3}$$

$$= \frac{(\ell_j + 2)(\ell_j + 1)/2}{3(\ell_j + 1)(\ell_j)/2} = \frac{\ell_j + 2}{3\ell_j} = 1 - \frac{2\ell_j - 2}{3\ell_j} < 1,$$

where the inequality follows from $\ell_j > 1$. Since V' has more entries with value 1, we conclude that the maximum value is reached when all entries assume value 1. Consequently, the worst case number of recurrences that need to be evaluated for filling a subtable D_v^i or D_v is $\mathcal{O}(3^\ell)$. The overall number of subtables to fill is $\mathcal{O}(\sum_{v \in V} \deg(v)) = \mathcal{O}(n)$. This implies the overall running time bound. □

3.2   An Extension to Subcases of List-Colored Graph Motif on Trees

The fixed-parameter tractability of GM on trees can be extended to give fixed-parameter tractability for LGM when the input graph G is a tree and the vertex-color graph H is a forest with bounded degree.

The first step in our algorithm is to apply the following two data reduction rules which are obviously correct.

Rule 1 *If there is a color vertex c in the vertex-color graph H such that $\deg_H(c) < M(c)$, then return "no".*

Rule 2 *If there is a color vertex c in the vertex-color graph H such that $\deg_H(c) = M(c)$, then set $\mathcal{L}(u) = \{c\}$ for all neighbors u of c in H.*

With these reduction rules at hand, we can show that the following special case of LGM is fixed-parameter tractable with respect to ℓ.

Lemma 1 LGM *can be solved in $\mathcal{O}(3^\ell \cdot n)$ time if G is a tree and the vertex-color graph H is a forest in which $\deg_H(c) - M(c) \leq 1$ for every color vertex c.*

Proof: We describe a reduction of this special case of LGM on trees to GM on trees. In the following, we assume without loss of generality that every color in the instance has multiplicity at least one in M. First, apply Reduction Rules 1 and 2 exhaustively. This can be done in $\mathcal{O}(n)$ time: First, we compute the vertex-color graph explicitly by traversing $\mathcal{L}(v)$ for each $v \in V$. In general, the size of these lists may exceed $\mathcal{O}(n)$ but, due to the following claim, we can immediately reject instances in which the sum of the list sizes is too big.

Claim: If the vertex-color graph H is a forest, then $\sum_{v \in V} |\mathcal{L}(v)| < 3n$ in any yes-instance. We show that H cannot have $3n$ edges. Since every entry of $\mathcal{L}(v)$ corresponds to an edge of H the claim follows. First, observe that in a yes-instance any of the n vertices from V has at most one leaf-neighbor from C in H. Thus, H contains at most n color vertices that are leaves. Moreover, if H contains $x \geq n$ non-leaf color vertices, then H contains a cycle: Consider the graph H' obtained from H by deleting all leaf-vertices that are color-vertices. The remaining color vertices have degree at least two in H'. Thus, H' has $2x \geq x + n$ edges and $x + n$ vertices and contains a cycle. Consequently, H has less than $3n$ vertices and thus if it it has $3n$ edges, then it contains a cycle, contradicting the assumption that H is a forest.

After building the vertex-color graph, we compute in $\mathcal{O}(n)$ time the difference between $M(c)$ and $\deg_H(c)$ once for all $c \in C$ and update this value whenever we delete an edge. Due to Reduction Rule 1, we have $M(c) \leq \deg_H(c)$ for every color vertex c in H. Moreover, since the reduction rules do not increase the degree of a vertex, we have $M(c) \geq \deg_H(c) - 1$. Finally, if $M(c) = \deg_H(c)$, then the connected component of c in H consists of c and $\deg_H(c)$ leaf neighbors of c. By the above, the vertex-color graph H contains the two types of connected components considered below. For both of them, we show that the constraints of $\mathcal{L}$ can be replaced by simple coloring constraints. Consider a connected component H' of H.

Case 1: H' is a star with a central color vertex c such that $M(c) = \deg_H(c)$ in M. Replace c by $\deg_H(c)$ new colors and assign a different color to each neighbor of c in H. This is equivalent as all neighbors of c in H are contained in any occurrence of M.

Case 2: H' is a tree in which each color vertex c fulfills $M(c) = \deg_H(c) - 1$. Let C' denote the set of color vertices in H' and V' denote the other vertices of H' that do not correspond to colors. Replace C' by one new color c^* and set the multiplicity of c^* to $|V'| - 1$. To show correctness of this replacement, we show that for every $v \in V'$, there is an assignment $f' : V' \setminus \{v\} \rightarrow C'$ such that $f'(u) \in \mathcal{L}(u)$ for each $u \in V' \setminus \{v\}$ and each color $c \in C'$ is assigned exactly $M(c) = \deg_H(c) - 1$ vertices by f'. To see the existence of this assignment consider the version of H' that is rooted at v. For each color vertex c in H', let V'_c denote the children of c in this rooted tree. For each vertex $u \in V'_c$ set $f'(u) := c$. With this assignment, there are exactly $\deg_H(c) - 1$ vertices that are assigned to c and every vertex $u \in V' \setminus \{v'\}$ is assigned to some color c of C', namely to its predecessor in H'. The construction of f' also shows that *exactly* one vertex of V' is not contained in any occurrence of M: deleting two or more vertices of V' implies that some color c of C' is assigned to less than

$M(c)$ vertices and deleting no vertex of V' implies that implies that some color c of C' is assigned to more than $M(c)$ vertices.

Applying these replacements exhaustively yields an equivalent instance of GM on trees which can be solved in the claimed running time due to Theorem 4. $\square$

We now show how to use the running time bound of Lemma 1 to obtain a fixed-parameter algorithm for the dual parameter ℓ for the special case of LGM when the color-vertex graph is a forest and when $M(C) := \max_{c \in C} M(c)$, the largest multiplicity in M, is bounded. The first step of the algorithm is to apply Reduction Rules 1 and 2 exhaustively. Afterwards, we branch on colors c where $\deg_H(c)$ exceeds $M(c)$ by at least two. We call such a color vertex *2-abundant* in the following.

Branching Rule 1 *If the vertex-color graph H contains a connected component H' with at least one 2-abundant color vertex, then do the following.*

- *Root H' arbitrarily.*

- *Choose some 2-abundant vertex c of H' such that the subtree of H' rooted at c has no further 2-abundant vertex.*

- *Choose a set V_c of $M(c) + 1$ arbitrary children of c.*

- *For each $u \in V_c$ branch into the case that c is removed from $\mathcal{L}(u)$.*

Proof of correctness: First, observe that such a 2-abundant color vertex c always exists and that it can be found in linear time by a bottom-up traversal of the rooted tree. Second, observe that since $M(c) \leq \deg_H(c) + 2$, the vertex c has at least $M(c) + 1$ children. Hence, if the original instance contains an occurrence of M, then there is some child u of c in H' which does not receive the color c in this occurrence. Thus, if the original instance is a yes-instance, then the branch in which we remove c from $\mathcal{L}(u)$ is a yes-instance. Conversely, any occurrence of M in an instance created by the branching rule is an occurrence of M in the original instance. $\square$

If Branching Rule 1 does not apply, then we can solve the instance in $\mathcal{O}(3^\ell \cdot n)$ time by Lemma 1. It thus remains to ensure that the rule cannot be applied too often. To this end, we apply one further reduction rule. To formulate the rule we need the following definition: we call a connected component H' of the vertex-color graph H *costly* if H' either consists of just one vertex $v \in V$ or H' is a tree such that all color vertices c in H' have multiplicity exactly $\deg_H(c) - 1$ in M.

Rule 3 *If G contains at least $\ell + 1$ costly components, then return "no".*

Proof of correctness: For each costly component at least one vertex is not contained in any occurrence of M. This is obvious for those components consisting only of one vertex v from V. For the other costly components, this follows from Case 2 in the proof of Lemma 1. $\square$

Now observe that in each instance created by an application of Branching Rule 1, the number of costly components is increased by exactly one: By the choice of c as a vertex without 2-abundant descendants in the rooted tree, the new connected component that is created by removing c from $\mathcal{L}(u)$ is either an isolated vertex or all color vertices have multiplicity exactly $\deg_H(c) - 1$. In both cases, the new component is costly. Hence, after at most $\ell + 1$ branching steps, Reduction Rule 3 directly reports that we have a no-instance. Since we branch into at most $M(C) + 1$ cases in each application of Branching Rule 1, we thus create $\mathcal{O}((M(C) + 1)^{\ell+1})$ instances that either adhere to the conditions of Lemma 1 or are rejected due to Reduction Rules 1 or 3 and can thus be solved in $\mathcal{O}(3^\ell \cdot n)$ time. Altogether, we obtain the following running time.

Theorem 5 *If G is a tree and the color vertex graph H is a forest, then* LGM *can be solved in $\mathcal{O}((M(C) + 1)^{(\ell+1)} \cdot 3^\ell \cdot n)$ time.*

When M is a set, the largest multiplicity is one, giving the following running time.

Corollary 2 *If G is a tree, H is a forest, and M is a set, then* LGM *can be solved in $\mathcal{O}(6^\ell \cdot n)$ time.*

By observing that Branching Rule 1 branches into at most $\deg_H(c) - 1$ branches, we also obtain the following running time bound in terms of the maximum degree of color vertices in H.

Corollary 3 *If G is a tree, and H is a forest whose color vertices have degree at most Δ_C, then* LGM *can be solved in $\mathcal{O}((\Delta_C - 1)^{(\ell+1)} \cdot 3^\ell \cdot n)$ time.*

3.3 A Kernelization Lower Bound

We now show that GM does not admit a polynomial-size problem kernel with respect to ℓ, even if G is a tree. The proof is based on a cross-composition [4] from the MULTICOLORED CLIQUE problem.

> MULTICOLORED CLIQUE
> **Input:** An undirected graph $H = (W, F)$ and a vertex-labeling $\lambda : W \to \{1, \ldots, k\}$.
> **Question:** Is there a vertex set $S \subseteq W$ such that $|S| = k$, the vertices in S have pairwise different labels, and $H[S]$ is a clique?

We refer to the colors of the MULTICOLORED CLIQUE instance as labels to avoid confusion with the colors of the GM instance. Informally, cross-compositions are reductions that combine many instances of one problem into one instance of another problem. The existence of a cross-composition from an NP-hard problem to a parameterized problem Q implies that Q does not admit a polynomial-size problem kernel (unless NP $\subseteq$ coNP/poly) [4].

Definition 3 ([4]) *Let $L \subseteq \Sigma^*$ be a language, let R be a polynomial equivalence relation on Σ^*, and let $Q \subseteq \Sigma^* \times \mathbb{N}$ be a parameterized problem. An or-cross-composition of L into Q (with respect to R) is an algorithm that, given t instances $x_1, x_2, \ldots, x_t \in \Sigma^*$ of L belonging to the same equivalence class of R, takes time polynomial in $\sum_{i=1}^{t} |x_i|$ and outputs an instance $(y, k) \in \Sigma^* \times \mathbb{N}$ of Q such that*

- *the parameter value k is polynomially bounded in $\max_{i=1}^{t} |x_i| + \log t$, and*

- *the instance (y, k) is a yes-instance for Q if and only if at least one instance x_i is a yes-instance for L.*

We present an or-cross composition of MULTICOLORED CLIQUE into GM on trees parameterized by ℓ. The polynomial equivalence relation R will be simply to assume that all the MULTICOLORED CLIQUE instances have the same number of vertices n. The main trick is to encode vertex identities in the graph of the MULTICOLORED CLIQUE instance by numbers of colored vertices in the GM instance; this approach was also followed in previous works on GM [12, 6].

Given t instances $(H_1 = (W_1, F_1), \lambda_1), \ldots, (H_t = (W_t, F_t), \lambda_t)$ of MULTICOL-ORED CLIQUE such that $|W_i| = n$ for all $i \in [t]$, we reduce to an instance of GM where the input graph is a tree as follows. Herein, we assume without loss of generality that $t = 2^s$ for some integer s.

The first construction step is to add one vertex r that connects the different parts of the instance and which will be contained in every occurrence of the motif. The vertex r thus receives a unique color that may not be deleted. To this vertex r we attach subtrees corresponding to edges of the input instances. Deleting vertices of such a subtree then corresponds to selecting the endpoints of the corresponding edge.

Instance selection gadget.   The technical difficulty in the construction is to ensure that the solution of GM deletes only vertices in subtrees corresponding to edges of the same graph. To achieve this, we introduce $k \cdot (k-1) \cdot \log t$ instance selection colors $\iota[p, q, \tau]$ where $p \in [k]$, $q \in [k] \setminus \{p\}$, and $\tau \in [\log t]$, and demand that the solution deletes exactly one vertex of each instance selection color. To ensure that exactly one instance is selected, we use two further colors ι^+ and ι^-.

For each MULTICOLORED CLIQUE instance (H_i, λ_i), attach an *instance selection path* P_i to r that is constructed based on the number i. Let $b(i)$ denote the binary expansion of i and let $b_\tau(i)$, $\tau \in [\log t]$, denote the τth digit of $b(i)$. Construct a path P_i containing first a vertex with color ι^+, then in arbitrary order exactly one vertex of each color in the color set $I_i := \{\iota[p, q, \tau] : b_\tau(i) = 1\}$, and then a vertex with color ι^-. Attach the path P_i to r by making the vertex with color ι^+ a neighbor of r.

The idea of the construction is that exactly one instance selection path P_i is completely deleted and that this will force any solution to delete paths that "complement" P_i (that is, paths which contain all $\iota[p, q, \tau]$ such that $b_\tau(i) = 0$) in the rest of the graph.

Edge selection gadget. To force deletion of subtrees corresponding to exactly $\binom{k}{2}$ edges with different labels, we introduce $2k(k-1)$ label selection colors $\lambda[p,q]^+$ and $\lambda[p,q]^-$ where $p \in [k]$ and $q \in [k] \setminus \{p\}$. These colors will ensure that, for each pair of labels p and q, the solution deletes exactly one path corresponding to the ordered pair (p,q) and one path corresponding to the pair (q,p).

There are two further sets of colors. One set is used for ensuring vertex consistency of the chosen edges, that is, to make sure that all the selected edges with label pair $(p, \cdot)$ correspond to the same vertex with label p. More precisely, we introduce a color $\omega[p,q]$ for each $p \in [k]$ and each $q \in [k] \setminus \{p\}$, except for the biggest $q \in [k] \setminus \{p\}$. The final color set is used to check that the edges selected for label pair (p,q) and for label pair (q,p) are the same. To this end, we introduce color $\varepsilon[p,q]$ for each $p \in [k]$ and each $q \in [k]$ such that $q > p$. To perform the checks of vertex and edge consistency, we encode the identities of vertices and edges into path lengths. More precisely, we assign each vertex $v \in W_i$ a unique (with respect to the vertices of W_i) number $\#(v) \in [n]$.

Now, for each label pair (p,q) and each instance i, attach one path $P_i(u,v)$ to r for each edge $\{u,v\}$ where u has color p and v has color $q \neq p$. The path $P_i(u,v)$

- starts with a vertex with color $\lambda[p,q]^+$ that is made adjacent to r,

- then contains exactly one vertex of each color in $\{\iota[p,q,\tau] : \iota[p,q,\tau] \notin I_i\}$,

- then contains $\#(u)$ vertices of color $\varepsilon[p,q]$ if $p < q$ and $n - \#(v)$ vertices of color $\varepsilon[q,p]$ if $p > q$,

- then, if q is not the biggest label in $[k] \setminus p$, contains $\#(u)$ vertices with color $\omega[p,q]$,

- then, if q is not the smallest label in $[k] \setminus p$, contains $n - \#(u)$ vertices with color $\omega[p,q']$, where q' is the next-smaller label in $[k] \setminus p$ (if $p = q - 1$, then $q' = q - 2$; otherwise $q' = q - 1$), and

- ends with a vertex with color $\lambda[p,q]^-$.

Let $\mathcal{C}$ denote the multiset containing all the vertex colors of all vertices added during the construction with their respective multiplicities. In the correctness proof it will be easier to argue about the colors that are *not* contained in M. Hence, the construction is completed by setting the multiset D of colors to "delete" to contain each color exactly once except

- the color of r which is not contained in D,

- the vertex consistency colors $\omega[p,q]$ each of which is contained with multiplicity n, and

- the edge selection colors $\varepsilon[p,q]$ each of which is contained with multiplicity n.

The motif M is defined as $M := \mathcal{C} \setminus D$. It remains to show the correctness.

Theorem 6 GRAPH MOTIF *does not admit a polynomial-size problem kernel with respect to ℓ even if G is a tree, unless* NP $\subseteq$ coNP/poly.

Proof: To complete the proof we need to show that the construction fulfills the properties of cross-compositions. First, the construction clearly runs in polynomial time. Second, the number of introduced colors is polynomially bounded in $k + \log t$ and thus the value of $\ell = |D|$ is polynomially bounded in $n + \log t$. Thus, it remains to show that the composition is an or-cross composition, that is:

> At least one (H_i, λ_i) is a yes-instance of MULTICOLORED CLIQUE $\Leftrightarrow$ $(M, G, \mathcal{L})$ is a yes-instance of GM.

($\Rightarrow$) Let $S \in W_i$ be a vertex set of size k such that $H_i[S]$ is a clique and the vertices in S have pairwise different labels. Consider the induced subgraph G' of G obtained by completely deleting the path P_i and, for each $\{u, v\} \in H_i[S]$, the paths $P_i(u, v)$ and $P_i(v, u)$. Since only complete paths are deleted and since each path in G is attached to r, the graph G' is connected. It remains to show that the multiset of deleted colors is D. First, ι^+ and ι^- are deleted once and contained once in D. Second, each instance selection color $\iota[p, q, \tau]$ is deleted once as required by D: If $\iota[p, q, \tau]$ is contained in P_i, then it is not contained in any $P_i(u, v)$. Conversely, if $\iota[p, q, \tau]$ is not contained in P_i then it is contained in each $P_i(u, v)$ where u has color p and v has color q. Third, exactly n vertices of each vertex consistency color $\omega[p, q]$ are deleted: these vertices are contained only in two paths $P_i(u, v)$, namely if u has label p and v has either label q or label $q + 1$. Since all the deleted paths with label pair $(p, \cdot)$ correspond to the same vertex u, the number of vertices with color $\omega[p, q]$ is $\#(v)$ if v has label q and $n - \#(v)$ if v has label p. Hence, exactly n vertices with this color are deleted, as required by D. Finally, we show that exactly n vertices of each edge selection color $\varepsilon[p, q]$, $p < q$, are deleted: Let u and v be the vertices of S with label p and q, respectively. Then, the deleted path $P_i(u, v)$ contains $\#(u)$ vertices with color $\varepsilon[p, q]$ and the deleted path $P_i(v, u)$ contains $n - \#(u)$ vertices with this color. Altogether, the multiset of colors in G' is exactly $\mathcal{C} \setminus D = M$.

($\Leftarrow$) Let G' be a connected subgraph of G whose multiset of vertex colors is exactly M. Let $V_D := V(G) \setminus V(G')$ denote the set of deleted vertices, that is, vertices not in G'. The color multiset of the vertex colors of V_D is exactly D. Thus, exactly one vertex with color ι^+ and one vertex with color ι^- are deleted. Consequently exactly one path P_i is completely deleted from G': deleting ι^+ in some P_i implies that the ι^- in P_i is also deleted. Thus, no further vertices from any P_j, $j \neq i$, may be deleted.

Moreover, since each label selection color $\lambda[p, q]^+$ or $\lambda[p, q]^-$ is contained exactly once in D, the set V_D also contains exactly one path $P_j(u, v)$ where u has label p and v has label q. Moreover, we have $j = i$ by the assignment of the instance selection colors: If $j \neq i$, then there is some $\tau \in [\log t]$ such that $b_\tau(i) \neq b_\tau(j)$. Then, however $\iota[p, q, \tau]$ is either not contained in the colors

of V_D or it is contained twice in the colors of V_D. In either case, the set of deleted colors is different from D.

Thus, all the deleted paths in the edge selection gadgets correspond to the same instance i. Now consider the paths for label pairs $(p, \cdot)$. These label pairs correspond to the same vertex: Otherwise, there is some $P_i(u, v)$ and some $P_i(u', v')$ such that $u \neq u'$, v has label q, and v' has label $q + 1$. Then, however, the number of vertices with color $\omega[p, q]$ does not equal n since $P_i(u, v)$ contains $\#(u)$ vertices of this color, $P_i(u', v')$ contains $n - \#(u')$ vertices of this color and $\#(u) \neq \#(u')$. Hence, the deleted paths correspond to a vertex set S with k different labels in some H_i. It remains to show that the graph $H_i[S]$ is a clique.

Consider an arbitrary pair of labels p and q where $p < q$. Moreover, let $u \in S$ and $v \in S$ have label p and q, respectively. Let $P_i(u, v')$ be the path for u that is deleted with this color pair and let $P_i(v, u')$ be the path for v that is deleted for this color pair. Then, $P_i(u, v')$ contains exactly $\#(u)$ vertices with color $\varepsilon[p, q]$, $P_i(v, u')$ contains exactly $n - \#(u')$ vertices of this color. Since D contains exactly n vertices of this color, this implies $u = u'$. By construction, this implies that u and v are neighbors in H_i. $\qquad\square$

4   Colorful Graph Motif on Trees

For the combination of vertex-colored trees as input graphs and motifs that are sets, the problem becomes considerably easier. First, we show that in this case CGM admits a linear-vertex problem kernel that can be computed in linear time. The problem kernelization is based on two simple observations. Recall that we call a vertex v unique if $\chi(v)$ occurs exactly once in G. Moreover, observe that in the case of CGM, the non-unique vertices are exactly those that are colored by abundant colors. The first observation is that the number of non-unique vertices is bounded by the dual parameter in CGM.

Lemma 2 *Let (M, G, χ) be an instance of* Colorful Graph Motif. *Then at most 2ℓ vertices in G are non-unique.*

Proof: Let C^+ denote the set of abundant colors, and let $\mathrm{occ}(c)$ denote the number of occurrences of a color c in G. Then, $n^+ := \sum_{c \in C^+} \mathrm{occ}(c)$ is the number of non-unique vertices. Let n^- denote the number of unique vertices in G. By definition, no color is repeated in M, thus $|M| = c^+ + n^-$. Moreover, $|V| = n^+ + n^-$. Hence, the number $\ell = |V| - |M|$ of vertices to delete satisfies $\ell = n^+ - c^+$. By definition of non-unique vertices we have $n^+ \geq 2c^+$, and thus we conclude that $\ell \geq n^+/2$. $\qquad\square$

Second, if the instance contains two unique vertices, then the uniquely determined path between these vertices is contained in every occurrence of the motif. The kernelization accordingly removes all the vertices that lie on these paths. More precisely, these vertices are "contracted" into the root r. Afterwards, in a second phase some further vertices are removed because their colors have been used
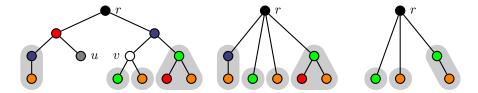
Figure 3: The two phases of the kernelization. Left: the input instance, where r, u, and v have unique colors; the pendant non-unique subtrees are highlighted by the gray background. Middle: after Phase I, all vertices on paths between unique vertices are contracted into r. Right: in Phase II, all vertices with a color that was removed in Phase I are removed together with their descendants.

during the contraction. Eventually, this results in an instance which has at most one unique vertex and thus, by Lemma 2, bounded size. For an example of the kernelization, see Figure 3. Below, we give a more detailed description.

Theorem 7 COLORFUL GRAPH MOTIF *on trees admits a problem kernel with at most $2\ell + 1$ vertices that can be computed in $\mathcal{O}(n)$ time.*

Proof: We first describe the kernelization algorithm, then we show its correctness and finally bound its running time. By Lemma 2, the size bound holds if the instance has no unique vertex. Thus, we assume that there is a unique vertex in the following.

Given an instance (M, G, χ) of CGM, first root the input tree G at an arbitrary unique vertex r. Now call a subtree with root v *pendant* if it contains all descendants of v in G. Then, compute in a bottom-up fashion maximal pendant subtrees such that no vertex in this subtree is unique. Call these subtrees the *pendant non-unique subtrees*. By Lemma 2, the total number of vertices in pendant non-unique subtrees is at most 2ℓ. Now the algorithm removes vertices in two phases.

Phase I. Remove from G all vertices except r that are not contained in a pendant non-unique subtree. Remove all colors of removed vertices from M. If there is a color c such that two vertices with color c are removed in this step, then return "no". Make r adjacent to the root of each pendant non-unique subtree.

Phase II. In the first step of this phase, for each color c where at least one vertex has been removed in Phase I, remove all vertices from G that have color c. In the second step of this phase, remove all descendants of these vertices. Finally, let M' denote the set of colors that are contained in the remaining instance. This completes the kernelization algorithm; the resulting instance has at most $2\ell + 1$ vertices since all vertices except r are unique. To show correctness, we first observe the following.

Claim: every occurrence of M in G contains no vertex v that is removed during Phase II of the kernelization. This can be seen as follows. First, every

occurrence of M in G contains all vertices removed during Phase I: these vertices are either unique or lie on the uniquely determined path between two unique vertices. Now consider a vertex v removed during Phase II. If v is removed in the first step of Phase II, then v has the same color c as a vertex u removed during Phase I. Consequently, v is not contained in an occurrence of M: By the above, the occurrence contains u and it contains no other vertex with color c. Otherwise, v is removed in the second step of Phase II, because v is not connected to r. Since every occurrence of M contains r, it thus cannot contain v.

We now show the correctness of the kernelization, that is, the equivalence of the original instance (M, G, χ) and the resulting instance (M', G', χ'). First, assume that (M, G, χ) is a yes-instance. Let S_T be an occurrence of M in G, and let T denote $G[S_T]$; by the above claim, T contains only vertices that are removed during Phase I or that are contained in G'. Consider the subtree T' of G that contains all vertices of T that are not removed during the kernelization. We show that T' is connected in G' and contains all colors of M'. Connectivity can be seen as follows. First, observe that T and T' contain r. Second, any vertex $v \neq r$ of T' is contained in some pendant non-unique subtree of G. Thus, v is in T connected to r via a path that first visits only vertices of T', including the root of the pendant non-unique subtree. The root of the pendant non-unique subtree is in G' adjacent to r. Thus, each vertex $v \neq r$ has in T' a path to r which implies that T' is connected. It remains to prove that T' contains all colors of M'. Consider a color $c \in M'$. Since $c \in M'$, none of the vertices with color c are removed in Phase I of the kernelization. Moreover, since no vertex of T is removed in Phase II of the kernelization, we have that the vertex of T with color c is contained in T'. Thus, T' contains each color of M'. Finally, T' contains each color at most once since T does.

Now assume that (M', G', χ') is a yes-instance and let $S_{T'}$ be an occurrence of M' in G'. Let T denote $G[S_{T'} \cup V_I]$, where V_I is the set of vertices removed during Phase I of the kernelization. We show that T is connected and contains every color of G exactly once. To see that T is connected observe the following: Clearly, $G[\{r\} \cup V_I]$ is connected. Moreover, each vertex $v \neq r$ of T' has in T' a path to r. This path contains a subpath from v to the root r' of the pendant non-unique subtree containing v. In G, r' is adjacent to some vertex of $\{r\} \cup V_I$. Therefore, r' is connected to r in T and thus T is connected. It remains to show that T contains every color of G exactly once. Clearly, T' contains at least one vertex of each color $c \in M'$. Moreover, it also contains at least one vertex of each color $c \in M \setminus M'$ since it contains all vertices of V_I. Besides, it contains each color only once: The vertices of T' have pairwise different colors and different colors than those of the vertices of V_I. Finally, the vertices of V_I have different pairwise colors since the kernelization did not return "no".

The running time can be seen as follows. Determining the pendant non-unique subtrees can be done by a standard bottom-up procedure in linear time. Removing all vertices during Phase I can also be achieved in linear time. After removing a vertex with color c in Phase I, we label c as *occupied*. When we remove a vertex with an occupied color during Phase I, we immediately return "no". After the removal of vertices during Phase I, we can construct M' from M

in linear time by removing each occupied color. Finally, we can in linear time add an edge between r and every root of a pendant non-unique subtree and then remove all remaining vertices that have an occupied color. The final graph G' is obtained by performing a depth-first search from r, in order to include only those vertices still reachable from r.   □

Now, let us turn to developing fast(er) FPT algorithms for CGM. It can be seen that it is possible to solve CGM in trees in time $1.62^\ell \cdot n^{\mathcal{O}(1)}$, by "branching on colors with the most occurrences" until every color appears at most twice. More precisely, for a color c that appears at least three times and some vertex v with color c, we can branch into the two cases to either delete v or to delete the at least two other vertices that have color c. The branching vector[2] for this branching rule is $(1, 2)$ or better. Now, if every color appears at most twice, then CGM on trees can be solved in polynomial time [12, Lemma 2]. By a different branching approach, the above running time can be further improved.

Branching Rule 2 *If there are at least two vertices u and v that are both not leaves of the tree G and the same color c, then branch into the case to delete from G either*

- *u and all vertices w such that the path from v to w contains u, or*

- *v and all vertices w such that the path from u to w contains v.*

Proof of correctness: In each case, a subtree is deleted. No occurrence may contain vertices of both subtrees, since then it contains u and v which have the same color.   □

If the rule does not apply, then one can solve the problem in linear time; here, let $\mathrm{occ}(c)$ denote the number of occurrences of a color c in G.

Lemma 3 *Let (M, G, χ) be an instance of* Colorful Graph Motif *such that G is a tree and for each color c with $\mathrm{occ}(c) > 1$ at least $\mathrm{occ}(c)-1$ occurrences of c are leaves of G, then (M, G, χ) can be solved in $\mathcal{O}(n)$ time.*

Proof: For each color c with $\mathrm{occ}(c) > 1$, the algorithm simply deletes $\mathrm{occ}(c) - 1$ leaves with color c. This can be done in linear time by visiting all leaves via depth-first search, checking for each leaf in $\mathcal{O}(1)$ time whether $\mathrm{occ}(c) > 1$ and deleting the leaf in $\mathcal{O}(1)$ time if this is the case. The resulting graph contains each color exactly once, and it is connected since a tree cannot be made disconnected by deleting leaves.   □

Altogether, we arrive at the following running time.

Theorem 8 Colorful Graph Motif *can be solved in $\mathcal{O}(\sqrt{2}^\ell + n)$ time if G is a tree.*

---

[2]For an introduction to the analysis of branching vectors, refer to [8, 16].

Proof: The algorithm is as follows. First, reduce the input instance in $\mathcal{O}(n)$ time to an equivalent one with $\mathcal{O}(\ell)$ vertices using the kernelization of Theorem 7. Now, apply Branching Rule 2. If this rule is no longer applicable, then solve the instance in $\mathcal{O}(\ell)$ time (by applying the algorithm behind Lemma 3). Since the graph has $\mathcal{O}(\ell)$ vertices, applicability of Branching Rule 2 can be tested in $\mathcal{O}(\ell)$ time. Thus, the overall running time is $\mathcal{O}(\ell)$ times the number of search tree nodes. Since each application of Branching Rule 2 creates two branches and reduces ℓ by at least two in each branch, the search tree has size $\mathcal{O}(2^{\ell/2}) = \mathcal{O}(\sqrt{2}^{\ell})$. The resulting running time is $\mathcal{O}(\sqrt{2}^{\ell} \cdot \ell + n)$. Furthermore, the factor of ℓ in the running time can be removed by interleaving search tree and kernelization [22], that is, by applying the kernelization algorithm of Theorem 7 in each search tree node. $\qquad\square$

5   Conclusion

In this paper, we focused on the GRAPH MOTIF, LIST-COLORED GRAPH MOTIF and COLORFUL GRAPH MOTIF problems, and in particular their behavior in terms of parameterized complexity, when the parameter is $\ell = |V| - |M|$, that is, the number of vertices of G that are not kept in a solution. We left open the parameterized complexity for parameter ℓ for LIST-COLORED GRAPH MOTIF on trees, even when the vertex-color graph is a forest.

As mentioned in the introduction, parameterization by ℓ may be interesting not only from a theoretic, but also from an applied point of view. Unfortunately, for the practically relevant case of LIST-COLORED GRAPH MOTIF we have obtained W[1]-hardness even for very restricted color lists $\mathcal{L}$. Moreover, as noted by Fertin et al. [14], a reduction of Rauf et al. [25] shows that the variant of COLORFUL GRAPH MOTIF where G is directed and has edge weights is W[1]-hard with respect to ℓ. However, the combination of ℓ with further structural parameters related to the colors of C led to tractability results [14, 15]. It would be interesting to identify such color-related structure also for LIST-COLORED GRAPH MOTIF.

References

[1] A. M. Ambalath, R. Balasundaram, C. R. H., V. Koppula, N. Misra, G. Philip, and M. S. Ramanujan. On the kernelization complexity of colorful motifs. In *Proceedings of the 5th International Symposium on Parameterized and Exact Computation (IPEC '10)*, volume 6478 of *Lecture Notes in Computer Science*, pages 14–25. Springer, 2010. `doi:10.1007/978-3-642-17493-3\_4`.

[2] N. Betzler, R. van Bevern, C. Komusiewicz, M. R. Fellows, and R. Niedermeier. Parameterized algorithmics for finding connected motifs in biological networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 8(5):1296–1308, 2011. `doi:10.1109/TCBB.2011.19`.

[3] A. Björklund, P. Kaski, and L. Kowalik. Constrained multilinear detection and generalized graph motifs. *Algorithmica*, 74(2):947–967, 2016. `doi:10.1007/s00453-015-9981-1`.

[4] H. L. Bodlaender, B. M. P. Jansen, and S. Kratsch. Kernelization lower bounds by cross-composition. *SIAM Journal on Discrete Mathematics*, 28(1):277–305, 2014. `doi:10.1137/120880240`.

[5] H. L. Bodlaender, S. Thomassé, and A. Yeo. Kernel bounds for disjoint cycles and disjoint paths. *Theoretical Computer Science*, 412(35):4570–4578, 2011. `doi:10.1016/j.tcs.2011.04.039`.

[6] É. Bonnet and F. Sikora. The Graph Motif problem parameterized by the structure of the input graph. *Discrete Applied Mathematics*, 231:78–94, 2017. `doi:10.1016/j.dam.2016.11.016`.

[7] S. Bruckner, F. Hüffner, R. M. Karp, R. Shamir, and R. Sharan. Topology-free querying of protein interaction networks. *Journal of Computational Biology*, 17(3):237–252, 2010. `doi:10.1089/cmb.2009.0170`.

[8] M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized Algorithms*. Springer, 2015. `doi:10.1007/978-3-319-21275-3`.

[9] B. Das, M. K. Enduri, N. Misra, and I. V. Reddy. On structural parameterizations of Graph Motif and Chromatic Number. In *Proceedings of the Third International Conference on Algorithms and Discrete Applied Mathematics (CALDAM '17)*, volume 10156 of *Lecture Notes in Computer Science*, pages 118–129. Springer, 2017. `doi:10.1007/978-3-319-53007-9\_11`.

[10] H. Dell and D. van Melkebeek. Satisfiability allows no nontrivial sparsification unless the polynomial-time hierarchy collapses. *Journal of the ACM*, 61(4):23:1–23:27, 2014. `doi:10.1145/2629620`.

[11] R. G. Downey and M. R. Fellows. *Fundamentals of Parameterized Complexity*. Springer, 2013. `doi:10.1007/978-1-4471-5559-1`.

[12] M. R. Fellows, G. Fertin, D. Hermelin, and S. Vialette. Upper and lower bounds for finding connected motifs in vertex-colored graphs. *Journal of Computer and System Sciences*, 77(4):799–811, 2011. `doi:10.1016/j.jcss.2010.07.003`.

[13] M. R. Fellows, D. Hermelin, F. Rosamond, and S. Vialette. On the parameterized complexity of multiple-interval graph problems. *Theoretical Computer Science*, 410(1):53–61, 2009. `doi:10.1016/j.tcs.2008.09.065`.

[14] G. Fertin, J. Fradin, and G. Jean. Algorithmic aspects of the Maximum Colorful Arborescence problem. In *Proceedings of the 14th Annual Conference on Theory and Applications of Models of Computation (TAMC '17)*, volume 10185 of *Lecture Notes in Computer Science*, pages 216–230, 2017. `doi:10.1007/978-3-319-55911-7\_16`.

[15] G. Fertin, J. Fradin, and C. Komusiewicz. On the Maximum Colorful Arborescence problem and color hierarchy graph structure. In *Proceedings of the 29th Annual Symposium on Combinatorial Pattern Matching (CPM '18)*, volume 105 of *LIPIcs*, pages 17:1–17:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018. `doi:10.4230/LIPIcs.CPM.2018.17`.

[16] F. V. Fomin and D. Kratsch. *Exact Exponential Algorithms*. Springer-Verlag, 1st edition, 2010. `doi:10.1007/978-3-642-16533-7`.

[17] R. Ganian. Twin-cover: Beyond vertex cover in parameterized algorithmics. In *Proceedings of the 6th International Symposium on Parameterized and Exact Computation (IPEC '11)*, volume 7112 of *Lecture Notes in Computer Science*, pages 259–271. Springer, 2011. `doi:10.1007/978-3-642-28050-4\_21`.

[18] S. Hartung, C. Komusiewicz, and A. Nichterlein. Parameterized algorithmics and computational experiments for finding 2-clubs. *Journal of Graph Algorithms and Applications*, 19(1):155–190, 2015. `doi:10.7155/jgaa.00352`.

[19] R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? *Journal of Computer and System Sciences*, 63(4):512–530, 2001. `doi:10.1006/jcss.2001.1774`.

[20] C. Komusiewicz, A. Nichterlein, R. Niedermeier, and M. Picker. Exact algorithms for finding well-connected 2-clubs in sparse real-world graphs: Theory and experiments. *European Journal of Operational Research*, 275(3):846–864, 2019. `doi:10.1016/j.ejor.2018.12.006`.

[21] V. Lacroix, C. G. Fernandes, and M.-F. Sagot. Motif search in graphs: Application to metabolic networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 3(4):360–368, 2006. `doi:10.1109/TCBB.2006.55`.

[22] R. Niedermeier and P. Rossmanith. A general method to speed up fixed-parameter-tractable algorithms. *Information Processing Letters*, 73(3-4):125–129, 2000. `doi:10.1016/S0020-0190(00)00004-1`.

[23] R. Y. Pinter, H. Shachnai, and M. Zehavi. Deterministic parameterized algorithms for the Graph Motif problem. *Discrete Applied Mathematics*, 213:162–178, 2016. `doi:10.1016/j.dam.2016.04.026`.

[24] R. Y. Pinter and M. Zehavi. Algorithms for topology-free and alignment network queries. *Journal of Discrete Algorithms*, 27:29–53, 2014. `doi:10.1016/j.jda.2014.03.002`.

[25] I. Rauf, F. Rasche, F. Nicolas, and S. Böcker. Finding maximum colorful subtrees in practice. *Journal of Computational Biology*, 20(4):311–321, 2013. `doi:10.1089/cmb.2012.0083`.

[26] R. Sharan and T. Ideker. Modeling cellular machinery through biological network comparison. *Nature Biotechnology*, 24(4):427–433, 2006. `doi:10.1038/nbt1196`.

[27] F. Sikora. An (almost complete) state of the art around the Graph Motif problem. Technical report, LIGM Université Paris-Est, March 2012. URL: `http://www.lamsade.dauphine.fr/~sikora/pub/GraphMotif-Resume.pdf`.