Upper and lower bounds for finding connected motifs in vertex-colored graphs

Michael R. Fellows, Guillaume Fertin, Danny Hermelin, Stéphane Vialette

Upper and Lower Bounds for Finding Connected Motifs in Vertex-Colored Graphs ⋆

Michael R. Fellows [a,1] Guillaume Fertin [b] Danny Hermelin [c,2] and Stéphane Vialette [d]

[a] *Office of DVC(Research), The University of Newcastle, Callaghan NSW 2308 - Australia*

[b] *Laboratoire d'Informatique de Nantes-Atlantique (LINA), UMR CNRS 6241 Université de Nantes, 2 rue de la Houssinière, 44322 Nantes Cedex 3 - France*

[c] *Algorithms and Complexity Group, Max Plank Institute for Informatics, Campus E1 4 66123, Saarbrücken - Germany*

[d] *IGM-LabInfo, CNRS UMR 8049, Université Paris-Est, 5 Bd Descartes 77454 Marne-la-Vallée - France*

Abstract

We study the problem of finding occurrences of motifs in vertex-colored graphs, where a motif is a multiset of colors, and an occurrence of a motif is a subset of connected vertices whose multiset of colors equals the motif. This problem is a natural graph-theoretic pattern matching variant where we are not interested in the actual structure of the occurrence of the pattern, we only require it to preserve the very basic topological requirement of connectedness. We give two positive results and three negative results that together give an extensive picture of tractable and intractable instances of the problem.

⋆ An extended abstract of this paper appeared in [23].

Email addresses: michael.fellows@newcastle.edu.au, fertin@lina.univ-nantes.fr, hermelin@mpi-inf.mpg.de, vialette@lri.fr (Stéphane Vialette).

[1] Supported by the Australian Research Council, by a Fellowship to the Durham University Institute for Advanced Studies, and by a William Best Fellowship at Grey College while the paper was in preparation.

[2] Most of the work on this is paper was done during the graduate studies of the

1 Introduction

Vertex-colored graph problems have numerous applications in bioinformatics. Sandwich problems have applications in DNA physical mapping [12,24,27] and in perfect phylogeny [14,33], while vertex-recoloring problems arise in protein-protein interaction networks and phylogenetic analysis [13,16,34]. In this paper, we consider another vertex-colored graph problem with an important application in metabolic network analysis [32]:

Graph Motif:

Input: A vertex-colored graph G and a multiset of colors M.

Question: Does G have a connected subset of vertices whose multiset of colors equals M?

The Graph Motif problem can be viewed as a natural variant of a graph-theoretic pattern matching problem. In more classical variants one is interested in subgraphs or induced subgraphs that appear, along with their colors, in the corresponding *text* graph. However, in Graph Motif we are not interested in the actual structure of the occurrence of the pattern, we only require it to preserve the very basic topological requirement of connectedness. We believe that due to the generic nature of this problem, it will find applications in many domains. Indeed, although the problem was introduced by Lacroix, Fernandes, and Sagot (in a slightly more general formulation) in the context of detecting patterns that occur in *e.g.* interaction networks between chemical compounds and\or reactions [32], it is conceivable to think that algorithms for this problem can also be used in analyzing other types of networks, such as social or technical networks.

In [32], Graph Motif is proved to be **NP**-complete even if the given vertex-colored graph is a tree, but fixed-parameter tractable in this case when parameterized by the size of the given motif (*i.e.*, $|M|$). However, as observed by [32], their fixed-parameter algorithm does not apply when the vertex-colored graph is a general graph. For this case they only provided a heuristic algorithm which works well in practice. This paper focuses on extending these results. In particular, we are interested in investigating Graph Motif under different parameters governing the structure of the input.

author at the University of Haifa. At the time, the author was supported by the Adams Fellowship of the Israel Academy of Sciences and Humanities.

1.1 Our results

We give an extensive, yet still initial, analysis for Graph Motif which provides both upper and lower bounds for the problem. Our analysis applies tools from both the classical and the parameterized complexity viewpoints, and is aimed at understanding the nature of tractable and intractable instances of the problem and the parameters controlling them. We show that Graph Motif is:

(1) **NP**-complete already for trees of maximum degree 3 and for motifs which are sets rather than multisets.
(2) **NP**-complete for motifs with 2 colors, even if G is bipartite with maximum degree 4.
(3) **FPT** when parameterized by the size of the motif k *via* an algorithm running in time $2^{O(k)}n^2 \lg n$.
(4) **XP** when parameterized by both the number of colors in the motif and the treewidth of the input graph. In other words, it is polynomial-time solvable when both these parameters are bounded by some constant.
(5) **W[1]**-hard when parameterized by the number of colors in the motif, even when the input graph is a tree.

Our first upper bound (3) should be compared with our first lower bound (1). These two together give a rather sharp distinction between the complexity of the problem when the motif is unbounded. When the motif has a bounded number of colors, our second lower bound (2) suggests that the problem remains hard for quite a few graph classes, yet not for bounded-treewidth graphs according to (4). Our last lower bound (5) complements the algorithm for bounded treewidth graphs by showing that when we take the number of colors in the motif as a parameter, the problem becomes intractable already for trees.

1.2 Related and extended work

Since the appearance of the earlier version of this work [23], other variants of Graph Motif have been introduced and studied. In [20], a minimization variant of the problem was considered, where the goal is to minimize the number of connected components that induce all the colors in the motif. An asymmetric maximization variant of Graph Motif was considered in [21], where the goal is to find the largest subset of the given motif that occurs in the graph. Both problems turn out to be **APX**-hard, even for restrictive special cases, and are in **FPT** when parameterized by the size of the solution by extensions of the algorithm presented here. Finally, Betzler *et al.* [8] considered

the variant where the occurrence of the motif is required to be 2-connected, and showed that this variant is **W[1]**-complete when parameterized by the size of the motif M. This result shows how fragile the tractability results are with respect to the topological requirements of the motifs. They also presented an improvement upon our **FPT** algorithm for parameter $|M|$, and this has recently been improved once more in [28].

The Graph Motif problem was introduced by by Lacroix, Fernandes, and Sagot (in a slightly more general formulation) in the context of detecting patterns that occur in various metabolic networks, *e.g.* interaction networks between chemical compounds and\or reactions [32]. There has been an enormous amount of study in this area in the biological community. We refer interested readers to [18] for a short overview. The model of Lacroix *et al.* was also used and extended in [15,29,30].

Finally, we wish to mention the related problem of Interval Constraint Coloring, where the input is a set of intervals on the integer line, and a set of motifs, one for each interval. The goal is to color the integers on the line so that each interval has exactly the colors of his motif. This problem was introduced in the context of automated mass spectrometry [2,3], and has been analyzed according to the parameters governing its input in [31].

1.3 Basic notation and terminology

Our graph-theoretic notation is standard and all notions we use can be found in any classical text on the subject, *e.g.* [19]. Throughout the paper, we use $G = (V(G), E(G))$ to denote our given vertex-colored graph, and $n = |V(G)|$ to denote its order. For a vertex $v \in V(G)$, we use $\chi(v)$ to denote the color of v, and for a vertex subset $V \subseteq V(G)$, we let $\chi(V)$ denote the multiset of colors $\bigcup_{v \in V} \chi(v)$. For any vertex subset $V \subseteq V(G)$, we let $G[V]$ denote the subgraph of G induced by V, *i.e.*, the subgraph on V along with all edges of G that connect vertices in V. We assume w.l.o.g. that G is connected.

A motif M is a multiset of colors. If M is in fact a set rather than a multiset, we say that M is *colorful*. Given a subset of vertices $V \subseteq V(G)$, $|V| = |M|$, we say that V is *colored by the colors of* M, if $\chi(V) = M$. For V to be an *occurrence* of M, we require not only for V to be colored by the colors of M, but also for $G[V]$ to be connected. If this is in fact the case, we say that M *occurs at* v for any vertex $v \in V$, or simply that M *occurs in* G. In these terms, the Graph Motif problem is the problem of determining whether a given motif M occurs in a given vertex-colored graph G. We assume w.l.o.g. that $\chi(v) \in M$ for any $v \in V(G)$.

Our analysis is based both on the classical and parameterized complexity

frameworks. Readers unfamiliar with these subjects are referred to [22,26].

1.4 Organization

The rest of the paper is organized as follows. In the reminder of this section we discuss notations that will be used throughout the paper. In Section 2, we give two **NP**-hardness results that will motivate the rest of our discussion. Following this, in Section 3 we present our algorithm for logarithmic size motifs. In Section 4 we discuss the case when G has bounded treewidth. Finally, in Section 5, we show that Graph Motif is **W[1]**-hard on trees when parameterized by the number of colors in M.

2 Two Classical Hardness Results

Graph Motif is known to be **NP**-complete for trees [32]. Our aim in this section is to tighten this result by showing that Graph Motif remains hard for highly restrictive graph classes, even if we restrict ourselves to motifs which are sets rather than multisets, or on the other extreme, to motifs which consist of a small number of colors. We begin with colorful motifs (*i.e.* set motifs).

Theorem 1 Graph Motif *is* **NP***-complete, even if M is colorful and G is a tree of maximum degree three.*

PROOF. Graph Motif is clearly in **NP**. To prove **NP**-hardness, we present a reduction from the well known **NP**-complete problem 3-SAT [26]. Recall that 3-SAT asks to determine whether a given 3-CNF formula is satisfiable, that is, whether there is a truth assignment to the boolean variables of the formula, such that the value of the formula under this assignment is 1. The problem remains hard even if each variable appears in at most three clauses and each literal (*i.e.*, variable with or without negation) appears in at most two clauses [26]. Hence, we restrict ourselves in our proof to formulas of this type.

Let an instance of 3-SAT be given in the form of a 3-CNF formula $\Phi = c_1 \wedge \cdots \wedge c_m$ over variables $x_1, \ldots, x_n$ such that $|\{c_j \mid x_i \in c_j\}| \leq 2$ and $|\{c_j \mid \bar{x}_i \in c_j\}| \leq 2$ for all $1 \leq i \leq n$. We construct an instance for Graph Motif as follows. The colored graph G initially consists of a path of n vertices, each colored by a distinct color in $1, \ldots, n$. To a vertex colored i in this path, $1 \leq i \leq n$, we connect a new vertex colored i'. To a vertex colored i', $1 \leq i \leq n$, we connect a pair of new non-adjacent vertices, both

colored x_i. Conceptually, each vertex in this pair corresponds to a different truth assignment for x_i. If a truth assignment to variable x_i satisfies clause c_j, we connect a new vertex colored c_j to the vertex colored x_i which corresponds to this assignment. This is done for every $x_i \in \{x_1, \ldots, x_n\}$ and every $c_j \in \{c_1, \ldots, c_m\}$. We conclude our construction by specifying M to be the set of colors $\{1, \ldots, n, 1', \ldots, n', x_1, \ldots, x_n, c_1, \ldots, c_m\}$. A simple example of this construction is given in Fig. 1. Note that G and M are as required by the theorem.

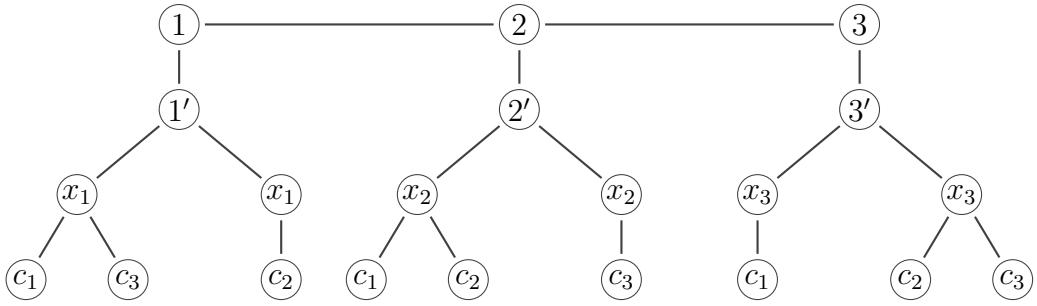

Figure 1. An example of the construction of G out of a 3-CNF formula which consists of three clauses: $c_1 = (x_1 \vee x_2 \vee x_3)$, $c_2 = (\bar{x}_1 \vee x_2 \vee \bar{x}_3)$, and $c_3 = (x_1 \vee \bar{x}_2 \vee \bar{x}_3)$.

The construction above is clearly polynomial. Hence, to complete the proof, we are left to show that M occurs in G if and only if Φ is satisfiable. For the first direction, assume that there exists a truth assignment ϕ which satisfies Φ. Let $N \subseteq V(G)$ be the subset of vertices in G which are colored by the colors in $\{1, \ldots, n, 1', \ldots, n'\}$, and let $X \subseteq V(G)$ be the subset of vertices which correspond to assignment ϕ. Hence, X consists of n vertices which are colored by the colors in $\{x_1, \ldots, x_n\}$, and $N \cup X$ induces a connected subgraph. Since ϕ satisfies every clause in Φ, by construction of G there is a vertex colored c_j in the neighborhood of X for every $1 \leq j \leq m$. In other words, there exists $C \subseteq N(X)$ which is colored by the colors in $\{c_1, \ldots, c_m\}$. It follows that $V = N \cup X \cup C$ is connected and is colored by the colors of M, and therefore is an occurrence of M in G.

For the converse direction, assume there exists an occurrence V of M in G. Let $X \subseteq V$ be the vertices colored by the colors in $\{x_1, \ldots, x_n\}$, $C \subseteq V$ be the vertices colored by the colors in $\{c_1, \ldots, c_m\}$, and ϕ the truth assignment corresponding to X. By construction, a vertex colored c_j is connected in G to a vertex colored x_i if and only if the truth assignment corresponding to this vertex satisfies clause c_j. Since C contains all colors in $\{c_1, \ldots, c_m\}$, and since vertices in C are connected only to vertices in X, it follows that ϕ satisfies every clause in Φ, and so it satisfies Φ itself. $\square$

Theorem 1 (and our proof) shows that a polynomial-time algorithm for Graph Motif restricted to trees of maximum degree three is unlikely, even when the

motif is colorful, and when each color occurs at most three times in the input tree. The next lemma shows that this implication is tight in two ways:

Lemma 2 GRAPH MOTIF *is polynomial-time solvable if either*

- *G has maximum degree two, or*
- *G is a tree where each color is shared by at most two vertices, and M is colorful.*

PROOF. First observe that if G has maximum degree two, then G is actually a collection of paths and cycles. Given a motif M, one can search through all subpaths of length $|M|$ in the cycles and paths of G in polynomial-time.

To prove the second part of the lemma, assume without loss of generality that G is rooted at $r \in V(G)$ (as otherwise we can try all possible roots), and let $M = \{1, \ldots, k\}$. We associate with each vertex v a unique literal $x(v) \in \{x_1, \bar{x}_1, \ldots, x_k, \bar{x}_k\}$, with $x(v) \in \{x_i, \bar{x}_i\}$ if $\chi(v) = i$. This can be done since each color in M appears at most twice in G. Next, we construct a 2-CNF formula Φ which consists of all clauses $x(u) \Rightarrow x(v)$ for every child-parent pair (u, v) in G. We also add to Φ a single-literal clause containing $x(r)$. Interpreting the assignment of the variable x_i as the selection of which of the two vertices colored i in G will be taken for the occurrence of M, it is easy to see that M occurs in G (with respect to the root r) iff Φ is satisfiable. Thus, the second part of the lemma follows from the fact that 2-SAT is polynomial-time solvable [7]. □

The motif used in the construction of Theorem 1 above is not only of unbounded size, it also consists of an unbounded number of colors. One might hope that for motifs which consist of only a small number of colors, GRAPH MOTIF would become polynomial-time solvable. Indeed, if the motif consists of a single color then GRAPH MOTIF reduces to the polynomial problem of computing the largest connected component in G. Unfortunately, the following theorem shows that already for motifs which consist of two colors GRAPH MOTIF is **NP**-complete, even when restricted to bipartite graphs of maximum degree four.

Theorem 3 GRAPH MOTIF *is* **NP***-complete, even if M consists of two colors, and G is bipartite with maximum degree four.*

PROOF. We reduce from the EXACT COVER BY 3-SETS (X3C) problem, which is known to be **NP**-complete [26]. Recall that, given a set $X = \{x_1, x_2, \ldots, x_{3q}\}$ and a collection $S = \{s_1, s_2, \ldots, s_n\}$ of 3-element subsets of X, the X3C problem asks to determine whether there exists an exact cover of

X in S, *i.e.*, a sub-collection $C \subseteq S$ such that every element of X is included in exactly one subset $s_i \in C$. The problem is hard even if each element of X appears in at most three sets of S [26], so we restrict ourselves in the proof to instances of this type.

Let $\langle X, S \rangle$ be an arbitrary instance of the X3C problem with $|\{s_j \in S \,|\, x_i \in s_j\}| \leq 3$ for all $x_i \in X$. We show how to construct a motif M and a colored graph G in such a way that there exists an exact cover of X in S if and only if M occurs in G. First, we define M so as it contains $2n + 3q$ *white* elements and q *black* elements. Then, we define G by $V(G) = X \cup S \cup S' \cup S''$ and $E(G) = E_1 \cup E_2 \cup E_3 \cup E_4$, where $S' = \{s'_1, s'_2, \ldots, s'_n\}$ and $S'' = \{s''_1, s''_2, \ldots, s''_n\}$ are copies of S, and E_1, E_2, E_3, E_4 are defined by: $E_1 = \{\{x_i, s_j\} \,|\, x_i \in s_j\}$, $E_2 = \{\{s_i, s'_i\} \,|\, 1 \leq i \leq n\}$, $E_3 = \{\{s'_i, s''_i\} \,|\, 1 \leq i \leq n\}$, and $E_4 = \{\{s''_i, s'_{i+1}\} \,|\, 1 \leq i \leq n-1\}$. The vertices of $X \cup S' \cup S''$ are colored white and the vertices of S are colored black. It is easily seen that G and M are as required by the theorem, and that our construction can be carried out in polynomial time.

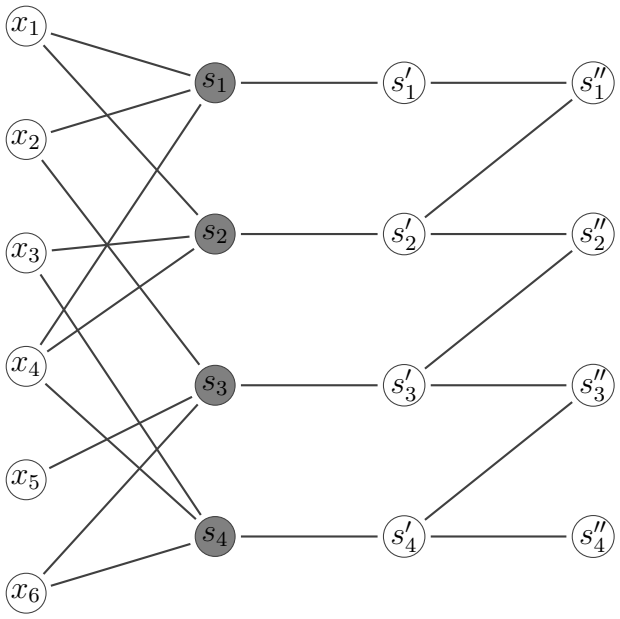

Figure 2. The construction of G out of an instance for X3C: $X = \{x_1, \ldots, x_6\}$ and $S = \{s_1, \ldots, s_4\}$. The 3-sets are $s_1 = \{x_1, x_2, x_4\}$, $s_2 = \{x_1, x_3, x_4\}$, $s_3 = \{x_2, x_5, x_6\}$, and $s_4 = \{x_3, x_4, x_6\}$.

Let us now argue that there exists an exact cover $C \subseteq S$ of X if and only if M occurs in G. For the first direction, suppose that there exists an exact cover $C \subseteq S$ of X. Consider the subset of vertices $V = X \cup C \cup S' \cup S''$. First note that V consists of $q = |C|$ black vertices and $2n + 3q = |X \cup S' \cup S''|$ white vertices. Second, since C is a cover of X, every vertex of X is connected to some vertex in C, and C is connected to $S' \cup S''$, so V itself is connected. It follows that V is an occurrence of M, and M occurs in G.

Conversely, suppose that there exists an occurrence $V \subseteq V(G)$ of M in G. Observe that M contains $2n + 3q$ white elements, and since exactly $2n + 3q$ vertices of G are colored white, we must have $X \cup S' \cup S'' \subset V$. The remaining q vertices in V are q black vertices from S. By construction, we do not have an edge between two vertices of X, nor between a vertex of X and a vertex of $S' \cup S''$. Therefore, since V is connected, each vertex of X has to be adjacent to at least one vertex in $V \cap S$. But $|X| = 3q$ and each vertex in S is connected to exactly 3 vertices in X. Then it follows that no two vertices of $V \cap S$ share a common neighbor in X, and $C = V \cap S$ is an exact cover of X in S. □

3 Parameterizing by the Motif Size

We next prove that Graph Motif is fixed-parameter tractable for parameter $|M|$ on any general graph. This generalizes the result of Lacroix, Fernandes, and Sagot [32] mentioned in Section 1, and should be compared to the hardness results presented in the previous section. In particular, our algorithm implies that Graph Motif is polynomial-time solvable when the given motif M is of logarithmic size (*i.e.* $|M| = O(\lg n)$).

Set $k := |M|$. Our algorithm for parameter k uses the *color-coding technique* introduced by Alon *et al.* [1], whose derandomized version crucially relies on the notion of *perfect hash families*: A family $\mathcal{F}$ of functions from $V(G)$ to $\{1, \ldots, k\}$ with the property that for any subset $V \subseteq V(G)$ of k vertices there is a function $f \in \mathcal{F}$ which is bijective when restricted to V. Alon *et al.* [1] showed any graph has a perfect hash family of size $2^{O(k)} \cdot \lg n$, for any given positive integer k, and furthermore, this family can be constructed in $2^{O(k)} \cdot n \lg n$ time.

The main idea of the color-coding technique is as follows: Suppose M has an occurrence V in G, and suppose we are provided with a perfect family $\mathcal{F}$ of hash functions from $V(G)$ to $\{1, \ldots, k\}$. Since $\mathcal{F}$ is perfect, we are guaranteed that at least one function in $\mathcal{F}$ assigns V with k distinct labels. Thus, by iterating through all functions in $\mathcal{F}$, we can concentrate on finding occurrences of V which are distinctly labeled by our hashing. This allows us to compute occurrences of M by applying dynamic programming.

Let us next describe the tables used in this dynamic programming algorithm, for a given a hash function $f : V(G) \to \{1, \ldots, k\}$. We use L to denote the set of labels $\{1, \ldots, k\}$. For each vertex $v \in V(G)$, we will have a table denoted DP_v. Each such table will have an entry for every pair of label-subset $L' \subseteq \{1, \ldots, k\}$, and motif-subset $M' \subseteq M$, $M' \neq \emptyset$. At the end of the

algorithm, the following invariant will be held:

$$\mathrm{DP}_v[L', M'] = 1 \iff M' \text{ has an occurrence } V \text{ with } v \in V, f(V) = L'. \quad (1)$$

All entries $\mathrm{DP}_v[\emptyset, M']$ are set to 0 at the beginning of the algorithm. Note that unlike M' which can be a multi-set, L' will always be a set of labels. Clearly, M has an occurrence V which is distinctly labeled by f iff $\mathrm{DP}_v[L, M] = v$ for some $v \in V(G)$.

Now let v be some vertex of G. Our goal is to compute $\mathrm{DP}_v[L', M']$ assuming $\mathrm{DP}_u[L'', M'']$ has previously been computed for every vertex $u \in V(G)$, any $L'' \subseteq L' \setminus \{f(v)\}$, and any $M'' \subseteq M' \setminus \{f(v)\}$. The straightforward approach is to consider small motifs occurring at neighbors of v. However, a motif occurring at v might be the union of motifs occurring at any number of neighbors of v, and so this approach might require exponential running time in n. We therefore present an alternative method for computing $\mathrm{DP}_v[L', M']$, which we call the *batch procedure*, that uses an even more naive approach, but one that requires exponential-time only with respect to k:

`Batch Procedure(`v, L', M'`):`

(1) Define $\mathcal{S}$ to be the set of all pairs (L'', M'') such that $L'' \subseteq L' \setminus \{f(v)\}$, $M'' \subseteq M' \setminus \{\chi(v)\}$, and $\mathrm{DP}_u[L'', M''] = 1$ for some neighbor u of v.
(2) Run through all pairs of $(L_1, M_1), (L_2, M_2) \in \mathcal{S}$ and determine whether $L_1 \cap L_2 = \emptyset$, and whether $M_1 \cup M_2 \subseteq M' \setminus \{\chi(v)\}$. If there is such a pair, add $(L_1 \cup L_2, M_1 \cup M_2)$ to $\mathcal{S}$ and repeat this step. Otherwise, continue to the next step.
(3) Set $\mathrm{DP}_v[L', M']$ to be 1 if $(L' \setminus \{\chi(v)\}, M' \setminus \{f(v)\}) \in \mathcal{S}$.

The reader should note that the correctness of the batch procedure relies on the fact that we are only required to compute motif occurrences which occur at specified vertices. In this way we are ensured that the connectedness requirement holds. This is stated more precisely in the proof of Lemma 4 below. Lemma 5 then shows that using the batch procedure, we can determine wether our given motif M occurs in G in relatively efficient (*i.e.* **FPT**-) time.

Lemma 4 *For any vertex $v \in V(G)$, label-subset $L' \subseteq L$, and $M' \subseteq M$, the batch procedure correctly computes $\mathrm{DP}_v[L', M']$, assuming $\mathrm{DP}_u[L'', M'']$ is given for every neighbor u of v, $L'' \subseteq L' \setminus \{f(v)\}$, and $M'' \subseteq M' \setminus \{\chi(v)\}$.*

PROOF. Let $\mathcal{S}$ be the family of pairs computed by the batch procedure. Consider any pair $(L'', M'') \in \mathcal{S}$ with $L'' = L' \setminus \{f(v)\}$. By construction, $M'' = M' \setminus \{\chi(v)\}$ and can be written as $M'' = M''_1 \cup \cdots \cup M''_\ell$, where each M''_i, $1 \leq i \leq \ell$, is a motif that has an occurrence V''_i which includes a neighbor of v. Furthermore, each V''_i is labeled by a unique set of labels L''_i such that

$L''_i \cap L''_j = \emptyset$ for all $1 \leq j \leq \ell$, $j \neq i$. It follows that all the V''_is are pairwise disjoint, and that $V' = \{v\} \cup V''_1 \cup \cdots \cup V''_\ell$ is connected. Hence, V' is an occurrence of $M' = M'' \cup \{\chi(v)\}$ in G which is labeled by $L' = L'' \cup \{f(v)\}$, and so invariant (1) is held when the batch procedure sets $\mathrm{DP}_v[L', M']$ to 1.

On the other hand, suppose M' has an occurrence V', $v \in V'$, which is labeled by L'. Write $M'' := M' \setminus \{\chi(v)\}$, $L'' := L' \setminus \{f(v)\}$, and $V'' := V' \setminus \{v\}$. Let $V''_1, \ldots, V''_\ell$ be the connected components of the induced subgraph $G[V'']$. Since V' is connected, every V''_i, $1 \leq i \leq \ell$, includes a neighbor of v. Furthermore, letting L''_i denote the set of labels that f assigns V''_i for every $1 \leq i \leq \ell$, we have $L' \subseteq L' \setminus \{f(v)\}$ and $L''_i \cap L''_j = \emptyset$ for all $1 \leq i, j \leq \ell$. It is now easy to see that the batch procedure will eventually compute the pair (M'', L'') in its second step, and hence $\mathrm{DP}_v[L', M']$ will be set to 1 in its final step. □

Lemma 5 *Given a labeling function $f : V(G) \to \{1, \ldots, k\}$, one can use the batch procedure iteratively in order to determine in $O(2^{6k}kn^2)$ time whether there is an occurrence of M in G which is distinctly labeled by f.*

PROOF. To prove the lemma, let us first analyze the complexity of a single invocation of the batch procedure. In its first step, the batch procedure searches through at most $2^k n$ motif families, each consisting of at most 2^k motifs. Hence, this step requires $O(2^{2k}kn)$ time. For the second step, notice that number of distinct motif and label-subset pairs is bounded by 2^{2k}, and so the number of times the second step is repeated is also bounded by this term. Since each iteration of this step can be computed in $O(2^{2k}k)$ time, it follows that the second step requires $O(2^{4k}k)$ time. Accounting also for the third step, the total time of one invocation of the batch procedure is therefore $O(2^{4k}k + 2^{2k}kn) = O(2^{4k}kn)$.

It now can easily be seen that due to Lemma 4, one needs to invoke the batch procedure at most $2^{2k}n$ times in order to compute every entry in each dynamic programming table DP_v. It follows that in $O(2^{6k}kn^2)$ time one can obtain all necessary information to determine whether M has an occurrence which is distinctly labeled by f, and so the lemma follows. □

Note that in case M is colorful, the vertex-coloring of G distinctly colors any occurrence of M, and therefore, in this case we can determine whether M occurs in G within the time complexity given in Lemma 5. For general motifs our algorithm is as follows: It first constructs in $2^{O(k)} \cdot n \lg n$ time a perfect family $\mathcal{F}$ of $2^{O(k)} \cdot \lg n$ functions from $V(G)$ to $\{1, \ldots, k\}$, using [1]. Next for each $f \in \mathcal{F}$, it uses the batch procedure to determine whether there is any occurrence of M which is distinctly labeled by f in $2^{O(k)} \cdot n^2$ time (Lemma 5). Since $\mathcal{F}$ is perfect, any occurrence of M in G is guaranteed to

be distinctly labeled by at least one labeling function $f \in \mathcal{F}$, and so by exhaustively searching through all functions in $\mathcal{F}$, our algorithm can determine whether M occurs in G in total $2^{O(k)} \cdot n^2 \lg n$ time.

Theorem 6 Graph Motif *can be solved in* $2^{O(k)} \cdot n^2 \lg n$ *time for* $k := |M|$.

4 Bounded Treewidth Graphs

The treewidth parameter of graphs [35] plays a central role in designing exact algorithms for many **NP**-hard graph problems (see *e.g.* [4,6,9,17]). In this section we show that Graph Motif is polynomial-time solvable when G has constant treewidth and M consists of a constant number of colors (but with arbitrary numbers of repeated elements in the multiset of colors). This should be compared with Theorem 3 which implies that Graph Motif is intractable even when M consists of only two colors and G is bipartite with constant degree.

4.1 Basic ingredients

We begin by describing the necessary terminology and definitions for our algorithm. We first define tree decompositions and nice tree decompositions of arbitrary graphs in a somewhat nonstandard way tailor-fit for our purposes. Later we introduce the central ingredient of our algorithm which is a data structure essential for its computation.

Definition 7 (Tree Decomposition [35]) *A* tree decomposition *of a graph G is a pair $(\mathcal{T}, \mathcal{X})$, where $\mathcal{X}$ is a set of subgraphs of G (we also refer to elements of $\mathcal{X}$ as bags), and $\mathcal{T}$ is a tree over $\mathcal{X}$, satisfying the following conditions:*

(1) $\bigcup_{X \in \mathcal{X}} X = G$, *and*
(2) $\mathcal{X}_v = \{X \in \mathcal{X} \mid v \in V(X)\}$ *is connected in $\mathcal{T}$ for all $v \in V(G)$.*

The width *of $\mathcal{T}$ is* $\max_{X \in \mathcal{X}} |V(X)| - 1$. *The* treewidth *of G is the minimum width over all tree decompositions of G.*

Definition 8 (Nice Tree Decomposition [11]) *A tree decomposition $(\mathcal{T}, \mathcal{X})$ of a graph G is nice if $\mathcal{T}$ is rooted, binary, and each bag in $\mathcal{X}$ is of one of the following four types:*

- *Leaf nodes $X \in \mathcal{X}$ are leaves in $\mathcal{T}$, and are singleton graphs which consist of a single isolated vertex $v \in V(G)$.*

- *Introduce nodes $X \in \mathcal{X}$ have one child X' in $\mathcal{T}$, with $V(X) = V(X') \cup \{v\}$ for some vertex $v \notin V(X')$, and $E(X) = E(X')$.*
- *Connect nodes $X \in \mathcal{X}$ have one child X' in $\mathcal{T}$, with $V(X) = V(X')$ and $E(X) = E(X') \cup \Big\{\{u, v\}\Big\}$ for some pair of vertices $u, v \in V(X)$.*
- *Forget nodes $X \in \mathcal{X}$ have one child X' in $\mathcal{T}$, with $V(X) = V(X') \setminus \{v\}$ for some vertex $v \in X'$, and $E(X) = E(X') \setminus \Big\{\{u, v\} \in E(X')\Big\}$.*
- *Join bags $X \in \mathcal{X}$ have two children X' and X'' in $\mathcal{T}$ with $X = X' = X''$.*

Although computing an optimal tree-decomposition of a graph is an **NP**-complete problem [5], for graph of constant treewidth there exists a linear-time algorithm due to Bodlaender [10] for computing an optimal tree-decomposition. Furthermore, given a tree decomposition for a given graph, one can obtain a nice tree decomposition with the same width (and with an at most constant-factor increase of nodes) in linear time (see for instance [11]). Therefore, we assume throughout the remainder of this section that we have a nice tree decomposition $(\mathcal{T}, \mathcal{X})$ of our input graph G.

Definition 9 ($X_{\mathcal{T}}$) *For any subgraph $X \in \mathcal{X}$, we let $X_{\mathcal{T}}$ denote the subgraph of G defined by*

$$X_{\mathcal{T}} = \bigcup_{X' \in Des(X)} X',$$

where $Des(X)$ is the collection of all descendants of X (including X itself) in $\mathcal{T}$.

Many of the treewidth-based algorithms for **NP**-complete graph problems apply some form of dynamic-programming computation on the nodes of the tree-decomposition at hand. Our algorithm also follows these lines, by maintaining for each $X \in \mathcal{X}$ the following data-structure:

Definition 10 (Descriptor, Inventory) *A descriptor of a subgraph X of G with respect to a motif M consists of:*

- *A partition $\Pi = \{\Pi_1, \ldots, \Pi_r\}$ of $V(X)$.*
- *A motif family $\mathcal{M} = \{M_1, \ldots, M_r\}$, where $M_i \subseteq M$, $M_i \neq \emptyset$, for all $1 \leq i \leq r$.*

We say that a descriptor as above is positive *for X if there exist r pairwise disjoint subsets of vertices in $X_{\mathcal{T}}$, say $V_1, \ldots, V_r$, such that:*

- *$V_i \cap V(X) = \Pi_i$, for all $1 \leq i \leq r$.*
- *V_i is an occurrence of M_i in $X_{\mathcal{T}}$, for all $1 \leq i \leq r$.*

We define the inventory *$inv(X)$ of X as the set of all positive descriptors for X.*

Lemma 11 *If G has constant treewidth and M consists of a constant number*

of colors, then any subgraph $X \in \mathcal{X}$ has a polynomial number of distinct positive descriptors.

PROOF. Suppose that any subgraph $X \in \mathcal{X}$ has at most α vertices, and that M consists of at most α colors, for some $\alpha \in \mathbb{N}^+$. The number of different vertex-partitions of any $X \in \mathcal{X}$ is bounded by α^α, and the number of different motif families consisting of at most α motifs, each of size at most n, is bounded by n^{α^2}. Therefore, the total number of positive descriptors for any $X \in \mathcal{X}$ is bounded by $\alpha^\alpha \cdot n^{\alpha^2}$. □

4.2 The algorithm

Our algorithm proceeds as follows. It traverses the nice tree-decomposition of G in bottom-up fashion, and for each node $X \in \mathcal{X}$ it encounters, it computes $inv(X)$ from the inventory (or inventories) of the child graph(s). We determine that M occurs in G iff $M \in inv(X)$ for some subgraph $X \in \mathcal{X}$. The following lemma proves the correctness of this strategy.

Lemma 12 *M occurs in G iff $M \in \mathcal{M}$ for some descriptor $(\Pi, \mathcal{M}) \in inv(X)$, $X \in \mathcal{X}$.*

PROOF. Suppose M has an occurrence V in G. For any $v \in V$, let $\mathcal{X}_v = \{X \in \mathcal{X} \mid v \in X\}$ denote the collection of subgraphs which include v. By property (2) of Definition 7, every $\mathcal{X}_v$ induces a connected subtree $\mathcal{T}_v$ in $\mathcal{T}$. Furthermore, property (1) of this definition implies that for any $u, v \in V$ with $\{u, v\} \in E(G)$, there is a subgraph in $\mathcal{X}_v$ that includes u, and a subgraph in $\mathcal{X}_u$ that includes v. It follows that $\{u, v\} \in E(G)$ implies $\mathcal{T}_u \cap \mathcal{T}_v \neq \emptyset$. Therefore, since V is connected, $\mathcal{T}_V = \bigcup_{v \in V} \mathcal{T}_v$ is a subtree of $\mathcal{T}$. Hence, there is a subgraph that lies on all paths in $\mathcal{T}$ between the root of $\mathcal{T}$ and $\{X_v \mid v \in V\}$. For this particular subgraph $X \in \{X_v \mid v \in V\}$, V is a subset of vertices in $X_{\mathcal{T}}$ with $V \cap V(X) \neq \emptyset$, and therefore there is some positive descriptor $(\Pi, \mathcal{M}) \in inv(X)$ with $M \in \mathcal{M}$. Since the converse direction is trivial, the lemma follows. □

We proceed to describe our algorithm in terms of the computation needed in order to obtain inventories of nodes in $\mathcal{T}$ from the inventories of their children. We use $(\Pi, \mathcal{M})$ to denote the descriptor of an arbitrary node X in $\mathcal{T}$, and $(\Pi', \mathcal{M}')$ to denote a descriptor $(\Pi, \mathcal{M})$ previously computed from the child X' of X in $\mathcal{T}$ (using double-prime notation in case X has two children). The base cases of this computation are the leaf nodes. The inventory of a leaf node X consists of a single descriptor $(\Pi, \mathcal{M})$ with $\Pi = \big\{\{v\}\big\}$ and $\mathcal{M} = \big\{\{\chi(v)\}\big\}$,

where v is the single vertex of X. The computation done on any single child node in $\mathcal{T}$ is almost equally easy, and is given for the sake of completeness in the following three lemmas.

Lemma 13 *Let X be an introduce node with child X' in $\mathcal{T}$ such that $V(X) = V(X') \cup \{v\}$ for some $v \notin V(X')$. Then $inv(X)$ is the set of all descriptors $(\Pi, \mathcal{M})$ with*

$$\Pi = \left\{\{v\}, \Pi'_1, \ldots, \Pi'_r\right\} \text{ and } \mathcal{M} = \left\{\{\chi(v)\}, M'_1, \ldots, M'_r\right\},$$

where $\Pi' = \{\Pi'_1, \ldots, \Pi'_r\}$ and $\mathcal{M}' = \{M'_1, \ldots, M'_r\}$ form a positive descriptor of X'.

PROOF. Clearly, $(\Pi, \mathcal{M})$ in the lemma is a positive descriptor of X, assuming $(\Pi', \mathcal{M}')$ is positive for X'. Furthermore, there are no other positive descriptors of X, since $X'_{\mathcal{T}}$ and $X_{\mathcal{T}}$ differ only by vertex v, and v is isolated in $X_{\mathcal{T}}$. □

Lemma 14 *Let X be a forget node with child X' in $\mathcal{T}$ such that $V(X) = V(X') \setminus \{v\}$ for some $v \in V(X')$. Then $inv(X)$ is the set of all descriptors $(\Pi' \setminus \Pi'_i, \mathcal{M}' \setminus M'_i)$ with $(\Pi', \mathcal{M}') \in inv(X')$ and $\Pi'_i = \{v\}$, and all descriptors $(\Pi, \mathcal{M})$ with*

$$\Pi = \left\{\Pi'_1, \ldots, \Pi'_i \setminus \{v\}, \ldots, \Pi'_r\right\} \text{ and } \mathcal{M} = \left\{M'_1, \ldots, M'_r\right\},$$

where $\Pi' = \{\Pi'_1, \ldots, \Pi'_r\}$ and $\mathcal{M}' = \{M'_1, \ldots, M'_r\}$ form a positive descriptor of X' with $\{v\} \subset \Pi'_i$.

PROOF. The lemma follows from the easy observation that any motif M'_i which occurs in $X'_{\mathcal{T}}$ occurs also in $X_{\mathcal{T}}$. Moreover, if V'_i is an occurrence of M'_i in $X'_{\mathcal{T}}$ with $V'_i \cap V(X') = X_i$, then V'_i is also an occurrence of M'_i in $X_{\mathcal{T}}$ with $V'_i \cap V(X) = \Pi'_i \setminus \{v\}$. □

Lemma 15 *Let X be a connect node with child X' in $\mathcal{T}$ such that $E(X) = E(X') \cup \left\{\{u, v\}\right\}$ for some $\{u, v\} \notin E(X')$. Then $inv(X)$ is the set of all descriptors in $inv(X')$, along with all descriptors $(\Pi, \mathcal{M})$ with*

$$\Pi = \left\{\Pi'_1, \ldots, \Pi'_{i-1}, \Pi'_i \cup \Pi'_j, \Pi'_{i+1}, \ldots, \Pi'_{j-1}, \Pi'_{j+1}, \ldots, \Pi'_r\right\} \text{ and}$$
$$\mathcal{M} = \left\{M'_0, \ldots, M'_{i-1}, M'_i \cup M'_j, M'_{i+1}, \ldots, M'_{j-1}, M'_{j+1}, \ldots, M'_r\right\},$$

where $\mathcal{M}' = (M'_1, \ldots, M'_r)$ and $\Pi' = (\Pi'_1, \ldots, \Pi'_r)$ form a positive descriptor of X', $\{u, v\} \nsubseteq \Pi'_i, \Pi'_j$ and $\{u, v\} \subseteq \Pi'_i \cup \Pi'_j$.

PROOF. First note that since any motif that occurs in $X'_{\mathcal{T}}$ occurs also in $X_{\mathcal{T}}$, and since $V(X') = V(X)$, we have $inv(X') \subseteq inv(X)$. Furthermore, any motif which occurs in $X_{\mathcal{T}}$ and not in $X'_{\mathcal{T}}$ can be decomposed into two motifs that occur at u and v. The lemma then follows. $\Box$

We are left to describe the more interesting case of join nodes. Let us first define the join operator $\otimes$ on two partitions of the same vertex set. Let Π' and Π'' be two partitions of the same vertex set. The partition $\Pi = \Pi' \otimes \Pi''$ corresponds to the equivalence relation formed by taking the transitive closure of $\Pi' \cup \Pi''$ (when Π' and Π'' are both viewed as equivalence relations). In other words, Π is the transitive closure of $\{(u, v) \mid u\Pi'v \text{ or } u\Pi''v\}$, where $u\Pi'v$ and $u\Pi''v$ respectively denote that u and v are in the same class in Π' and in Π''.

Now, for a pair of motif families $\mathcal{M}' = (M'_1, \ldots, M'_p)$ and $\mathcal{M}'' = (M''_1, \ldots, M''_q)$ associated as in Definition 10 with the partitions Π' and Π'', we define the motif family $\mathcal{M} = \mathcal{M}' \otimes \mathcal{M}'' = (M_1, \ldots, M_r)$, such that M_i is defined by adding and subtracting colors in the natural manner. That is, if $\Pi'_{i_1}, \ldots, \Pi'_{i_x} \in \Pi'$ and $\Pi''_{i_1}, \ldots, \Pi''_{i_y} \in \Pi''$ are the classes of vertices forming $\Pi_i \in \Pi$, then the motif M_i is defined by

$$M_i = \Big(\bigcup_{j \in \{1,\ldots,x\}} M'_{i_j} \setminus \chi(\Pi'_{i_j}) \Big) \quad \bigcup \quad \Big(\bigcup_{j \in \{1,\ldots,y\}} M''_{i_j} \setminus \chi(\Pi''_{i_j}) \Big) \quad \bigcup \quad \chi(\Pi_i). \qquad (2)$$

Lemma 16 *Let X be a join node with children X' and X'' in $\mathcal{T}$. Then $inv(X)$ is the set of all descriptors in $inv(X')$ and $inv(X'')$, along with all descriptors $(\Pi' \otimes \Pi'', \mathcal{M}' \otimes \mathcal{M}'')$ with $(\Pi', \mathcal{M}') \in inv(X')$ and $(\Pi'', \mathcal{M}'') \in inv(X'')$.*

PROOF. First observe that $inv(X') \subseteq inv(X)$ and $inv(X'') \subseteq inv(X)$ follow immediately from the definition. To show that any descriptor $(\Pi' \otimes \Pi'', \mathcal{M}' \otimes \mathcal{M}'')$ with $(\Pi', \mathcal{M}') \in inv(X')$ and $(\Pi'', \mathcal{M}'') \in inv(X'')$ is also in $inv(X)$ consider an arbitrary class $\Pi_i \in \Pi$ along with its corresponding motif $M_i \in \mathcal{M}$, and let $\Pi'_{i_1}, \ldots, \Pi'_{i_x} \in \Pi'$ and $\Pi''_{i_1}, \ldots, \Pi''_{i_y} \in \Pi''$ be the classes of vertices from which Π_i originated, and $M'_{i_1}, \ldots, M'_{i_x}$ and $M''_{i_1}, \ldots, M''_{i_y}$ be the motifs corresponding to these vertex classes in $inv(X')$ and $inv(X'')$ respectively. We will argue that the motif $M_i \in \mathcal{M}$ has an occurrence V_i with $V_i \cap V(X) = \Pi_i$.

To see this, take $V'_{i_1}, \ldots, V'_{i_x}$ and $V''_{i_1}, \ldots, V''_{i_y}$ to be the occurrences of $M'_{i_1}, \ldots, M'_{i_x}$ and $M''_{i_1}, \ldots, M''_{i_y}$ in $X'_{\mathcal{T}}$ and $X''_{\mathcal{T}}$, respectively, and define $V_i \subseteq V(X_{\mathcal{T}})$ by

$$V_i = V'_{i_1} \cup \cdots \cup V'_{i_x} \cup V''_{i_1} \cup \cdots \cup V''_{i_y}.$$

Then V_i is colored by the colors of M_i according to (2), and by the requirements of $V'_{i_1}, \ldots, V'_{i_x}$ and $V''_{i_1}, \ldots, V''_{i_y}$ according to Definition 10. Furthermore, $V_i \cap V(X) = \Pi_i$. Now, for any pair of vertices $u, v \in V(X)$, if u and v are in

the same class in Π' or Π'', then u and v are connected in $X_{\mathcal{T}}[V_i]$, since they are even connected in a subset of V_i. Thus, as Π is the transitive closure of $\Pi' \cup \Pi''$, it follows that all vertex-pairs in Π_i are connected in $X_{\mathcal{T}}[V_i]$. As all vertex subsets $V'_{i_1}, \ldots, V'_{i_x}, V''_{i_1}, \ldots, V''_{i_y}$ are connected in $X_{\mathcal{T}}[V_i]$ and have non-empty intersection with Π_i, there is a path from any vertex in V_i to Π_i. Combining the two facts above together we get that V_i is connected in $X_{\mathcal{T}}$.

To prove the converse direction of the lemma, we show that for any positive descriptor $(\Pi, \mathcal{M})$ of X' which is not in the inventories of X' and X'', there are two descriptors $(\Pi', \mathcal{M}') \in inv(X')$ and $(\Pi'', \mathcal{M}'') \in inv(X'')$ with $\Pi = \Pi' \otimes \Pi''$ and $\mathcal{M} = \mathcal{M}' \otimes \mathcal{M}''$. first observe any motif $M_i \in \mathcal{M}$ that does not occur in $X'_{\mathcal{T}}$ nor in $X''_{\mathcal{T}}$, has an occurrence V_i which can be decomposed into its connected components $V'_{i_1}, \ldots, V'_{i_x}$ in $X'_{\mathcal{T}}[V_i]$, and into the connected components $V''_{i_1}, \ldots, V''_{i_y}$ in $X''_{\mathcal{T}}[V_i]$. These vertex-subsets are all occurrences of sub-motifs of M_i, and moreover, the families $\{V'_{i_1} \cap V(X), \ldots, V'_{i_x} \cap V(X)\}$ and $\{V''_{i_1} \cap V(X), \ldots, V''_{i_y} \cap V(X)\}$ both partition $V_i \cap V(X)$. Taking the transitive closure of the union of both these partitions, we must get $V_i \cap V(X)$, as otherwise V_i cannot be connected in $X_{\mathcal{T}}$. Continuing in this way for all motifs in $\mathcal{M}$, we get that $\Pi = \Pi' \otimes \Pi''$ and $\mathcal{M} = \mathcal{M}' \otimes \mathcal{M}''$. □

Theorem 17 *If G has constant treewidth and M has a constant number of colors, then* Graph Motif *is polynomial-time solvable.*

PROOF. The algorithm is as described below. Given an instance (G, M) of Graph Motif, it first computes in linear time a nice tree-decomposition $(\mathcal{T}, \mathcal{X})$ of G, and then it traverses $\mathcal{T}$ in bottom-up fashion, computing the inventories of each subgraph $X \in \mathcal{X}$ it encounters from the inventories of its child subgraph(s) in $\mathcal{T}$. The algorithm then determines that M occurs in G iff M is in a motif family of a positive descriptor of some $X \in \mathcal{X}$. Correctness of this approach follows from Lemma 12, and from the correctness of lemmas 13 through 16. Furthermore, lemmas 13 through 16 together imply that computing the inventory of any subgraph requires at most quadratic time in the total size of the inventory of its children. Hence, as each inventory consists of a polynomial number of descriptors (according to Lemma 11), this can be performed in polynomial-time. Therefore, as $|\mathcal{X}| = \mathcal{O}(n)$ nodes, the entire algorithm requires polynomial-time. □

5 Parameterizing by the Number of Colors

Although Theorem 17 gives a nice complementary result to the sharp hardness result of Theorem 3, it still leaves a certain gap. In the following section we aim to close this gap, by proving that Graph Motif, parameterized by the

number of colors c in M, is **W[1]**-hard for trees. We refer readers to [22,25] for more information on the **W[1]** class, and the implications of an **FPT**-time algorithm for a **W[1]**-hard problem.

Theorem 18 *The* Graph Motif *problem, parameterized by the number of colors c in the motif, is* **W[1]***-hard for trees.*

PROOF. We present a parameterized reduction from Clique which is known to be **W[1]**-hard [22]. Recall that in Clique we are given a simple graph H and an integer κ, the parameter, and we are asked to determine whether H has a subset of κ vertices which are pairwise adjacent. Given an instance $\langle H, \kappa \rangle$ of Clique, we describe a rooted tree $G = T$ colored with $1 + \kappa + 2\kappa(\kappa - 1) + \binom{\kappa}{2}$ colors. We let m denote the number of edges of H, *i.e.*, $m = |E(H)|$.

- The root of T is colored a.
- The root has $\kappa \cdot |V(H)|$ children organized into κ groups $S(1) \dots S(\kappa)$. The group of $|V(H)|$ children $S(i)$ consists of the nodes $s(i, u)$, where $u \in V(H)$. The color of each node in $S(i)$ is $b(i)$.
- From each node $s(i, u)$ hang $\kappa - 1$ groups of paths. The groups are $P(i, u, j)$ for every $j \in \{1 \dots, \kappa\} \setminus \{i\}$. There is one path $p(i, u, j, v) \in P(i, u, j)$ for each edge $\{u, v\} \in E(H)$ that is incident to u in H.

The path $p(i, u, j, v)$ begins with a vertex colored $c(i, j)$ and ends with a vertex colored $d(i, j)$, and otherwise consists of some number $m(i, u, j, v)$ of internal vertices colored by $e(i, j) = e(j, i)$. There is an important detail to note here. If $i < j$, then $c(i, j)$ and $c(j, i)$ are different colors, and likewise $d(i, j)$ and $d(j, i)$, whereas $e(i, j)$ and $e(j, i)$ are the same color. We call the $e(i, j)$'s the *edge colors*.

The number $m(i, u, j, v)$ is calculated as follows. Number the edges in $E(H)$ from 1 to m, letting $l(uv)$ denote the number assigned to the edge $\{u, v\} \in E(H)$. We define:

$$m(i, u, j, v) = \begin{cases} l(uv) & \text{if } i < j \\ 3m - l(uv) & \text{if } i > j. \end{cases}$$

The motif M consists of one-of-each for every color other than the edge colors, and $3m$ elements colored by each edge color. Thus, M consists of $c = 1 + \kappa + 2\kappa(\kappa - 1) + \binom{\kappa}{2}$ different colors, and $|M| = 1 + \kappa + 2\kappa(\kappa - 1) + 3m\binom{\kappa}{2}$. This completes the construction of our instance $\langle (G, M), c \rangle$ for Graph Motif. Illustrations of the construction are given in Figures 3 and 4.

We claim that H has a subset of κ pairwise adjacent vertices if and only if T has a subtree T' which is an occurrence of M. Suppose that the vertices

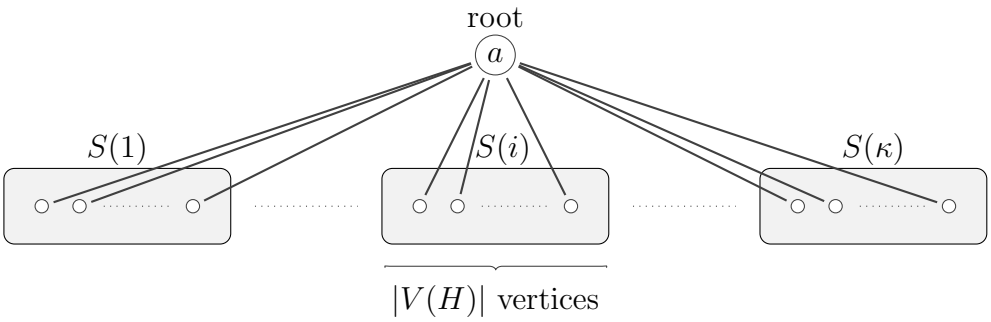

Figure 3. A schematic description of the the first two levels of the tree.

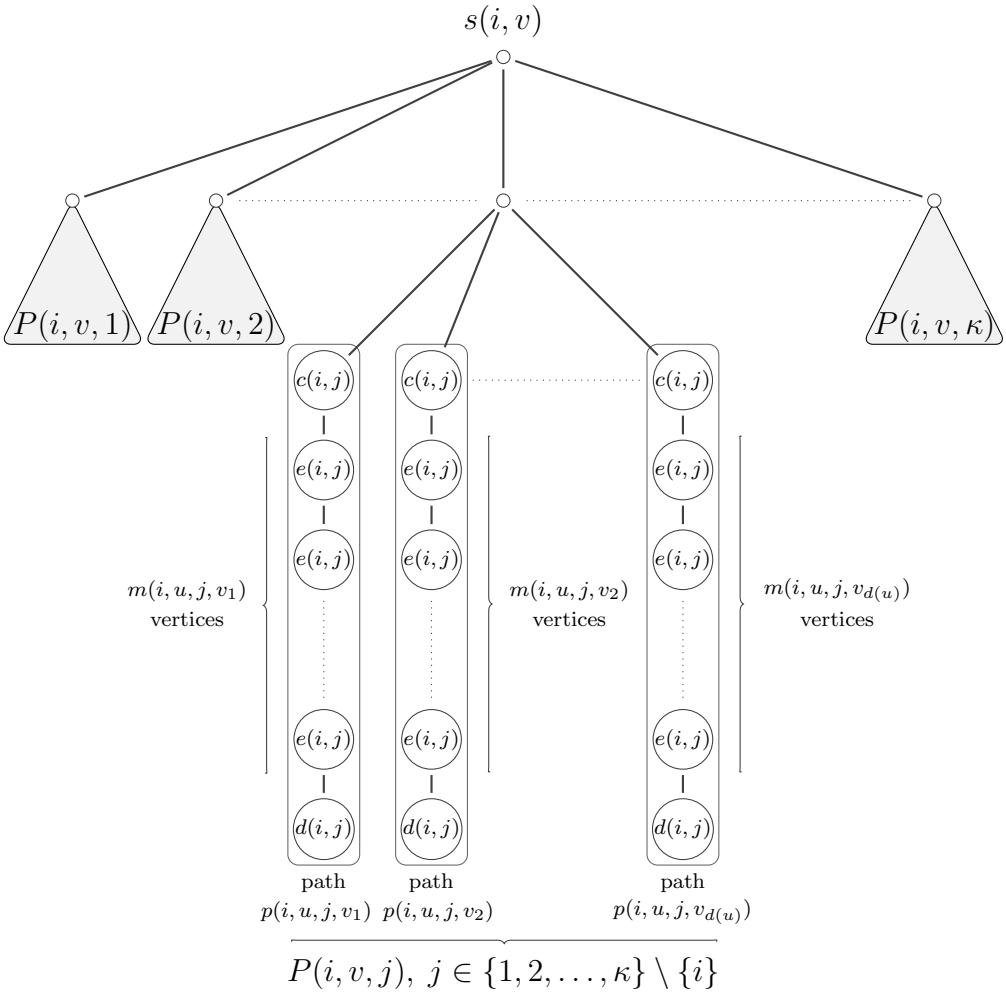

Figure 4. A schematic description of the group of paths which hang from the node $s(i,v) \in S(i)$. Is is assumed here that $N(u) = \{v_1, v_2, \ldots, v_{d(u)}\}$.

$v_1, ..., v_\kappa$ are pairwise adjacent in H. The subtree T' consists of:

- The root which is colored a.
- The κ children of the root $s(i, v_i)$ for all $1 \leq i \leq \kappa$, where $s(i, v_i)$ is colored $b(i)$.

- The $\kappa(\kappa-1)$ paths $p(i, v_i, j, v_j)$. The path $p(i, v_i, j, v_j)$ begins with a node colored $c(i, j)$ and ends with a node colored $d(i, j)$ for all $1 \leq i, j \leq \kappa$, $i \neq j$. Note that the path $p(i, v_i, j, v_j)$ is pendant from $s(i, v_i)$ since v_i and v_j are adjacent in H. Together, the two complementary paths $p(i, v_i, j, v_j)$ and $p(j, v_j, i, v_i)$ contain $3m$ nodes colored $e(i, j)$.

In the other direction, suppose that the subtree T' of T is an occurrence of M. Then T' must include the root of T, since it is the only node colored a. Furthermore, T' must contain exactly one node in each of the groups $S(i)$, $1 \leq i \leq \kappa$, since nodes in each $S(i)$ are all colored $b(i)$. Suppose these nodes are $s(1, v_1), \ldots, s(\kappa, v_\kappa)$. We argue that the vertices $v_1, ..., v_\kappa$ are pairwise adjacent in H.

In order for T' to be an occurrence of M in T, T' must contain exactly one pendant path in each of the groups of paths $P(i, v_i, j)$ for any $1 \leq i, j \leq \kappa$, $i \neq j$, and nothing further. To see this, note that T' must contain at least one path in each of the groups of paths $P(i, v_i, j)$ in order for T' to contain a node colored $d(i, j)$. But containing one such path prevents T' from including any node of other paths in $P(i, v_i, j)$, else T' would contain too many nodes of color $c(i, j)$.

It follows that for any pair of indices i, j with $1 \leq i < j \leq \kappa$, T' includes exactly two paths $p(i, v_i, j, x)$ and $p(j, v_j, i, y)$ that contain nodes of color $e(i, j) = e(j, i)$. Since M contains exactly $3m$ elements colored by $e(i, j)$, it follows that $x = v_j$ and $y = v_i$, since $p(i, v_i, j, v_j)$ and $p(j, v_j, i, v_i)$ are the only two paths in T with nodes colored $e(i, j)$ that together have exactly $3m$ nodes of this color. But then, by the construction of T, v_i and v_j must be adjacent in H. □

6 Summary

In this paper we studied the Graph Motif problem, a natural pattern-matching problem in graphs which was introduced by Lacroix, Fernandes, and Sagot to model applications in metabolic network analysis [32]. We presented an extensive complexity analysis of the problem which entailed two positive results, and three negative results. Our analysis focused on finding natural parameters that determine the complexity of the problem. To this extent, we identified two such natural parameters: The size of the motif, and the number of different colors in the motif. We believe however that there are more parameters to be explored, and identifying these is left for further research.

Another direction for possible future research is classes of dense graphs. The reader should also note that all of our hardness results use graphs that are quite sparse, which makes the task of assuring connectivity of potential motif

occurrences difficult. On the other hand, Graph Motif is trivial on the complete graph. We conjecture that there are many other natural and interesting dense graph classes for which Graph Motif becomes tractable.

Finally, we conclude with a list of three concrete questions that were left open by this paper:

- Is Graph Motif in **FPT** when parameterized by the treewidth of the graph, and the number of different colors in the motif is taken as a constant?
- In light of Theorem 1, Theorem 18, and Lemma 2, is Graph Motif restricted to trees fixed-parameter tractable when parameterized by both the number of colors of the motif, and the maximum number of occurrences of a color in the tree?
- Does Graph Motif have a polynomial-size kernel?

References

[1] N. Alon, R. Yuster, and U. Zwick. Color coding. *Journal of the ACM*, 42(4):844–856, 1995.

[2] E. Althaus, S. Canzar, K.M. Elbassioni, A. Karrenbauer, and J. Mestre. Approximating the interval constrained coloring problem. In *Proceedings of the 11th Scandinavian Workshop on Algorithm Theory (SWAT)*, pages 210–221, 2008.

[3] E. Althaus, S. Canzar, M.R. Emmett, A. Karrenbauer, A.G. Marshall, A. Meyer-Bäse, and H. Zhang. Computing H/D-exchange speeds of single residues from data of peptic fragments. In *Proceedings of the 23rd ACM Symposium on Applied Computing (SAC)*, pages 1273–1277, 2008.

[4] S. Arnborg. Efficient algorithms for combinatorial problems on graphs with bounded decomposability. A survey. *BIT Numerical Mathematics*, 25(1):2–23, 1985.

[5] S. Arnborg, D.G. Corneil, and A. Proskurowski. Complexity of finding embeddings in a k-tree. *SIAM Journal on Algebraic and Discrete Methods*, 8(2):277–284, 1987.

[6] S. Arnborg and A. Proskurowski. Linear time algorithms for NP-hard problems restricted to partial k-trees. *Discrete Applied Mathematics*, 23:11–24, 1989.

[7] B. Aspvall, M.F. Plass, and R.E. Tarjan. A linear-time algorithm for testing the truth of certain quantified boolean formulas. *Information Processing Letters*, 8:121–123, 1979.

[8] N. Betzler, M.R. Fellows, C. Komusiewicz, and R. Niedermeier. Parameterized algorithms and hardness results for some graph motif problems. In *Proceedings*

of the 19th annual symposium on Combinatorial Pattern Matching (CPM), pages 31–43, 2008.

[9] H.L. Bodlaender. A tourist guide through treewidth. *Acta Cybernetica*, 11:1–23, 1993.

[10] H.L. Bodlaender. A linear time algorithm for finding tree-decompositions of small treewidth. *SIAM Journal on Computing*, 25:1305–1317, 1996.

[11] H.L. Bodlaender. Treewidth: Algorithmic techniques and results. In *Proceedings of the 22nd international symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 19–36, 1997.

[12] H.L. Bodlaender and L.E. de Fluiter. Intervalizing k-colored graphs. In *Proceedings of the 22nd International Colloquium on Automata, Languages and Programming (ICALP)*, pages 87–98, 1995.

[13] H.L. Bodlaender, M.R. Fellows, M.A. Langston, M.A. Ragan, F.A. Rosamond, and M. Weyer. Kernelization for convex recoloring. In *Proceedings of the 2nd workshop on Algorithms and Complexity in Durham (ACiD)*, pages 23–35, 2006.

[14] H.L. Bodlaender, M.R. Fellows, and T. Warnow. Two strikes against perfect phylogeny. In *Proceedings of the 19th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 273–283, 1992.

[15] S. Bruckner, F. Hüffner, R.M. Karp, R. Shamir, and R. Sharan. Torque: topology-free querying of protein interaction networks. *Nucleic Acids Research*, 37:106–108, 2009.

[16] B. Chor, M.R. Fellows, M.A. Ragan, F.A. Rosamond, and S. Snir. Connected coloring completion for general graphs: Algorithms and complexity. In *Proceedings of the 13th conference on computing and combinatorics (COCOON)*, pages 75–85, 2007.

[17] D.G. Corneil and J.M. Keil. A dynamic programming approach to the dominating set problem on k-trees. *SIAM Journal on Algebraic and Discrete Methods*, 8(4):535–543, 1987.

[18] Y. Deville, D. Gilbert, J. Van Helden, and S.J. Wodak. An overview of data models for the analysis of biochemical pathways. *Briefings in Bioinformatics*, 4(3):246–259, 2003.

[19] R. Diestel. *Graph Theory*. Springer-Verlag, New York, 2000.

[20] R. Dondi, G. Fertin, and S. Vialette. Weak pattern matching in colored graphs: Minimizing the number of connected components. In *Proceedings of the 10th Italian Conference on Theoretical Computer Science (ICTCS)*, pages 27–38, 2007.

[21] R. Dondi, G. Fertin, and S. Vialette. Maximum motif problem in vertex-colored graphs. In *Proceedings of the 20th annual symposium on Combinatorial Pattern Matching (CPM)*, pages 221–235, 2009.

[22] R. Downey and M. Fellows. *Parameterized Complexity*. Springer-Verlag, 1999.

[23] M.R. Fellows, G. Fertin, D. Hermelin, and S. Vialette. Sharp tractability borderlines for finding connected motifs in vertex-colored graphs. In *Proceedings of the 34th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 340–351, 2007.

[24] M.R. Fellows, M.T. Hallett, and H.T. Wareham. DNA physical mapping: Three ways difficult. In *Proceedings of the 1st annual European Symposium on Algorithms (ESA)*, pages 157–168, 1993.

[25] J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006.

[26] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, New York, 1979.

[27] M. Golumbic, H. Kaplan, and R. Shamir. On the complexity of DNA physical mapping. *Advances in Applied Mathematics*, 15:251–261, 1994.

[28] S. Guillemot and F. Sikora. Finding and counting vertex-colored subtrees – Manuscript. 2010.

[29] T. Ideker, R.M. Karp, J. Scott, and R. Sharan. Efficient algorithms for detecting signaling pathways in protein interaction networks. *Journal of Computational Biology*, 13(2):133–144, 2006.

[30] B.P. Kelley, R. Sharan, R.M. Karp, T. Sittler, D.E. Root, B.R. Stockwell, and T. Ideker. Conserved pathways within bacteria and yeast as revealed by global protein network alignment. *Proceedings of the National Academy of Sciences of the United States of America*, 100(20):11394–11399, 2003.

[31] C. Komusiewicz, R. Niedermeier, and J. Uhlmann. Deconstructing intractability: A case study for interval constrained coloring. In *Proceedings of the 20th annual symposium on Combinatorial Pattern Matching (CPM)*, pages 207–220, 2009.

[32] 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.

[33] F.R. McMorris, T.J. Warnow, and T. Wimer. Triangulating vertex-colored graphs. *SIAM Journal on Discrete Mathematics*, 7(2):296–306, 1994.

[34] S. Moran and S. Snir. Convex recolorings of strings and trees: Definitions, hardness results and algorithms. In *Proceedings of the 9th international Workshop on Algorithms and Data Structures (WADS)*, pages 218–232, 2005.

[35] N. Robertson and P.D. Seymour. Graph minors. II. Algorithmic aspects of tree-width. *SIAM Journal of Algorithms*, 7:309–322, 1986.