Parameterized Algorithmics for Finding Connected Motifs in Biological Networks

Nadja Betzler, René van Bevern, Michael R. Fellows, Christian Komusiewicz, and Rolf Niedermeier

Abstract—We study the NP-hard LIST-COLORED GRAPH MOTIF problem which, given an undirected list-colored graph $G = (V, E)$ and a multiset M of colors, asks for maximum-cardinality sets $S \subseteq V$ and $M' \subseteq M$ such that $G[S]$ is connected and contains exactly (with respect to multiplicity) the colors in M'. LIST-COLORED GRAPH MOTIF has applications in the analysis of biological networks. We study LIST-COLORED GRAPH MOTIF with respect to three different parameterizations. For the parameters motif size $|M|$ and solution size $|S|$ we present fixed-parameter algorithms, whereas for the parameter $|V|-|M|$ we show W[1]-hardness for general instances and achieve fixed-parameter tractability for a special case of LIST-COLORED GRAPH MOTIF. We implemented the fixed-parameter algorithms for parameters $|M|$ and $|S|$, developed further speed-up heuristics for these algorithms, and applied them in the context of querying protein-interaction networks, demonstrating their usefulness for realistic instances. Furthermore, we show that extending the request for motif connectedness to stronger demands such as biconnectedness or bridge-connectedness leads to W[1]-hard problems when the parameter is the motif size $|M|$.

Index Terms—parameterized complexity, color-coding, list-colored graphs, pattern matching in graphs, protein-interaction networks

1 INTRODUCTION

WITH the advent of network biology [1, 29] and complex network analysis in general, the study of pattern matching problems in graphs has become more and more important. In this context, the term "graph motif" plays a central role. Roughly speaking, there are two views of graph (or network) motifs. The older is the topological view primarily concerned with subgraph isomorphism problems. For instance, the term "network motif" has been used to represent patterns of interconnections that occur in a network at frequencies much higher than those found in random networks [30, 32]. In contrast, the second and more recent view on graph motifs takes a more "functional approach" [9, 18, 24]. Here, topology is of lesser importance, while the functionalities of network nodes (expressed by colors) form the governing principle.

In this work, we concentrate on the second, functional view of graph motifs. Typically, functional categories that are associated to network vertices are represented as "colors". Hence, we consider *list-colored graphs* which means that each vertex v is associated with a set of colors $\mathrm{col}(v)$. A formal definition of the main problem studied in this work, following Lacroix et al. [24], is:

> LIST-COLORED GRAPH MOTIF:
> *Input:* A list-colored undirected graph $G = (V, E)$ and a multiset of colors M.
> *Task:* Find maximum-cardinality sets $S \subseteq V$ and $M' \subseteq M$ such that the induced subgraph $G[S]$ is connected and there is a one-to-one mapping f from S to M' such that $\forall v \in S : f(v) \in \mathrm{col}(v)$.

Fellows et al. [18] studied a restricted variant of this problem, GRAPH MOTIF, which is the special case of LIST-COLORED GRAPH MOTIF where each vertex is associated with a single color, that is, $|\mathrm{col}(v)| = 1$ for all $v \in V$. These types of graphs will be called *vertex-colored* (instead of list-colored) in this work. An even further restricted variant of GRAPH MOTIF demands that the motif is a set instead of a multiset. We call motifs that are sets *colorful*.

Next, we briefly describe two applications for LIST-COLORED GRAPH MOTIF in the area of biological network analysis.

Reaction Motifs in Metabolic Networks

The LIST-COLORED GRAPH MOTIF problem was originally defined in the context of finding motifs in metabolic networks [24]. Here, the motif is a multiset of reaction types and the given network is a reaction graph: vertices correspond to reactions, and two vertices are connected if the corresponding reactions can occur successively, that is, the products of one reaction are the inputs of the other reaction or vice versa. The task is to find a subgraph of the reaction graph that is connected and has exactly the reaction types specified by the motif. Since a specific reaction may be classified as an instance of more than one reaction

N. Betzler, R. van Bevern, C. Komusiewicz, and R. Niedermeier are with Institut für Informatik, Friedrich-Schiller-Universität Jena, Ernst-Abbe-Platz 2, D-07743 Jena, Germany. E-mail: {nadja.betzler, rene.bevern, c.komus, rolf.niedermeier}@uni-jena.de.
M.R. Fellows is with the School of Engineering and Information Technology, Charles Darwin University, Darwin, Northern Territory 0909, Australia. E-Mail: michael.fellows@newcastle.edu.au.
Some of the results of this work appeared under the title "Parameterized Algorithms and Hardness Results for Some Graph Motif Problems" in the proceedings of the 19th Annual Symposium on Combinatorial Pattern Matching (CPM '08), 2008 [4]. While the conference version mainly deals with motif problems on vertex-colored graphs, here the focus is on the more general list-colored graphs. We now also study two further parameterizations and perform experimental studies.

type, the vertices of the reaction graph may be colored by more than one color resulting in a list-colored graph.

Topology-free Querying of Protein-interaction Networks

Querying of networks is the task of identifying a subnetwork of a large network that is similar (both topologically and functionally) to a given small network (the query). A prominent special case of this problem is the task of finding signalling paths in protein-interaction networks that are similar to a given path [22, 23, 27]. Extensions of this approach can handle queries that are trees or tree-like [6, 15]. Bruckner et al. [9] studied the querying problem when the topology of the query is unknown. The query is a set of proteins that is known to form a protein complex in some species A and the task is to identify similar complexes in a different species B. This can be done by searching in the protein-interaction network of species B for a subnetwork that is similar to the query. More specifically, every protein of the query is identified by one color and a protein in the interaction network receives the colors of all query proteins with similar sequence (that is, the BLAST score exceeds a a predefined threshold). The main point is that, since the topology of the query is unknown, no assumptions about the topology of the subnetwork that corresponds to this protein complex are made, with one exception: the subnetwork should be connected or almost connected. Furthermore, of all subnetworks meeting these criteria, the one with maximum-weight spanning tree should be reported: The edge weights in the network correspond to interaction probabilities. Accordingly, a spanning tree with large weight corresponds to a high interaction probability among the proteins of the putative complex. This leads to a weighted version of LIST-COLORED GRAPH MOTIF with the further restriction that the motif is a set because each protein of the query is identified with a unique color.

Known Results

We first summarize the current state of the art concerning the algorithmic complexity of GRAPH MOTIF, that is, the special case where G is vertex-colored. GRAPH MOTIF is computationally hard. Formulated as a decision problem, it is NP-complete even if the input graph is a tree [24]. This can be strengthened to an NP-completeness result even when the input multiset M actually is a set and the input graph is a tree with maximum vertex degree three [18]. Moreover, NP-completeness has also been shown for the case that M consists of only two colors and the input graph is restricted to be bipartite with maximum degree four [18]. Given the apparent hardness of GRAPH MOTIF, Fellows et al. [18] initiated a parameterized complexity analysis showing that GRAPH MOTIF, parameterized by the motif size $|M|$ is fixed-parameter tractable. The algorithm with the asymptotically fastest running time for GRAPH MOTIF is due to Guillemot and Sikora [20], who use algebraic techniques to solve both the problem of finding an occurrence of a motif, and the problem of counting the number of occurrences. Dondi et al. [13] extended these investigations for GRAPH MOTIF by studying the case where the subgraph induced by the chosen motif vertices does not need to be connected. Furthermore, GRAPH MOTIF is hard in terms of approximability and fixed-parameter tractability with respect to the parameter "solution size" $|S|$ [14]. The algorithm with the best asymptotic running time for GRAPH MOTIF and parameter $|S|$ runs in $O(4^{|S|} \cdot |S|^2 \cdot m)$ time on m-edge graphs [20].

Concerning LIST-COLORED GRAPH MOTIF, there are fewer results, although the hardness results for GRAPH MOTIF clearly hold as well. Bruckner et al. [9] presented a fixed-parameter algorithm for LIST-COLORED GRAPH MOTIF with a worst-case running time of $O(|M|! \cdot 3^{|M|} \cdot m)$. They also showed that on many real-world instances the actual running time is much smaller than this worst-case estimate, and successfully applied their algorithm to protein-interaction networks. Furthermore, a web-server for solving LIST-COLORED GRAPH MOTIF (relying on fixed-parameter algorithms and an integer linear programming formulation) is available [8]. Blin et al. [6] presented a further algorithm for LIST-COLORED GRAPH MOTIF, based on a 0/1-integer programming formulation.

Our Results

This work presents fixed-parameter algorithms as well as parameterized hardness results for several variants of LIST-COLORED GRAPH MOTIF. In Section 3, we consider different parameterizations for LIST-COLORED GRAPH MOTIF. First, considering the parameter $|M|$, we present an algorithm that solves LIST-COLORED GRAPH MOTIF in $O(10.88^{|M|} \cdot m)$ time on m-edge graphs. Second, we consider the parameter solution size $|S|$. Adapting an approach of Dondi et al. [14] for GRAPH MOTIF, we present a randomized algorithm for LIST-COLORED GRAPH MOTIF that runs in $O(29.6^{|S|} \cdot |S| \cdot m)$ time. Third, we consider the dual parameter $n - |M|$, that is, the minimum number of vertices that are *not* in a solution. We show that in general LIST-COLORED GRAPH MOTIF becomes W[1]-hard[1] for this parameterization, and that for the special case where M is colorful and G is vertex-colored, LIST-COLORED GRAPH MOTIF can be solved in $O(2^{n-|M|} \cdot m)$ time.

We implemented our two fixed-parameter algorithms for LIST-COLORED GRAPH MOTIF parameterizing in the one case by $|M|$ and in the other case by $|S|$. We developed and implemented a number of further heuristic speed-ups of both algorithms, and we report here on their performance for real-world instances.[2] As one example of such a heuristic speed-up, we identify cases in which applying a different color-coding procedure helps in decreasing the running time. Furthermore, we identify

1. This excludes hope for fixed-parameter tractability with respect to this parameter [16, 19, 25].

2. Source code available under http://theinf1.informatik.uni-jena.de/graph-motif/

cases in which the application of a simple (randomized) brute force algorithm yields a speed-up.

We applied our algorithms in the context of querying of protein-interaction networks, showing that both the algorithm with parameter $|M|$ and the algorithm with parameter $|S|$ can solve almost all of the considered input instances within 10 minutes, and that in most cases the algorithm with parameter $|M|$ outperforms the algorithm with parameter $|S|$. Furthermore, we examined the quality of the solutions by testing them for enrichment of functional annotation terms: for each solution, which is a set of proteins, we retrieved functional annotation terms from public databases [3, 28, 31] and applied the GO::TermFinder tool [7] to find annotation terms that have a statistically significant overrepresentation compared to random protein sets. We found that about 78% have a significant enrichment in at least one functional annotation and that 100% of the solutions of size at least four show this enrichment. Furthermore, the percentage of solutions that have this functional enrichment is roughly the same for solutions with $|S|/|M| \geq 40\%$, and is lower when $|S|/|M| < 40\%$.

Finally, we further chart the range of (theoretical) tractability of LIST-COLORED GRAPH MOTIF by exploring what happens if we demand "more" than simple connectivity. We show that if one requires that the found motif shall not only be connected but biconnected or bridge-connected (see Section 2 for formal definitions), then, in both cases, the corresponding LIST-COLORED GRAPH MOTIF problem becomes W[1]-complete with respect to the parameter motif size. Since these are the two simplest ways of demanding more than connectivity, this shows that the request for connected motifs is already a topology demand close to the border between tractability and intractability. These W[1]-hardness results also generalize to higher connectivity demands such as p-connectivity and p-edge connectivity (see Section 2 for formal definitions). Even further, W[1]-hardness also holds for uncolored graphs, where one searches for a subgraph with the specific connectivity demand, and the parameter is the number of subgraph vertices.

Somewhat aside, we study a further variant of GRAPH MOTIF, the MIN-CC GRAPH MOTIF problem, which, given a vertex-colored graph G and a multiset of colors M, asks for a set of vertices of G that has the colors of M and induces a graph with at most d connected components. We answer an open question of Dondi et al. [13] by showing that MIN-CC GRAPH MOTIF is W[1]-hard with respect to d, even if the input graph is restricted to be only a path.

2 PRELIMINARIES

We only consider simple undirected graphs $G = (V, E)$, where $n := |V|$ and $m := |E|$. For a vertex set $S \subseteq V$ we denote by $G[S] := (S, \{\{u, v\} \mid \{u, v\} \in E \wedge u, v \in S\})$ the subgraph of G *induced by* S. A *list-coloring* of an undirected graph $G = (V, E)$ is a function $\mathrm{col} : V \to 2^C$, where C is a set of colors, that is, each vertex of G is associated with a set of colors $\mathrm{col}(v)$. In case $|\mathrm{col}(v)| = 1$ for all $v \in V$, we say that G is *vertex-colored* and denote by $\mathrm{col}(v)$ the (uniquely determined) color of v. Unless stated otherwise, a *motif* is a multi-set of colors. In case the motif is a set, we call the motif *colorful*. An *occurrence* of a motif M in G is a set of vertices $S \subseteq V$ such that $|S| = |M|$, $G[S]$ is connected, and there are x vertices of color c in S if and only if M contains c exactly x times. A vertex u in an undirected graph is called a *cut-vertex* if there are two vertices v, w with $v \neq u$ and $w \neq u$ such that every path from v to w contains u. If an undirected graph G is connected and has no cut-vertex, then G is *biconnected*. In general, if a graph $G = (V, E)$ cannot be disconnected by deletion of any set of p vertices, it is called *p-connected*. A *bridge* in an undirected graph is an edge $\{u, v\}$ such that every path between u and v contains $\{u, v\}$. If G is connected and has no bridge, then G is *bridge-connected*. A graph is called *p-edge-connected* if it cannot be disconnected by deletion of any set of p edges.

We briefly introduce the relevant notions of parameterized complexity theory [16, 19, 25]. Parameterized algorithmics aims at a multivariate (at least two-dimensional) complexity analysis of problems (also see the recent surveys [17, 26]). The hope lies in confining the seemingly inevitable combinatorial explosion of NP-hard problems to a parameter k. A given parameterized problem (I, k) is *fixed-parameter tractable (FPT)* with respect to the parameter k if it can be solved within running time $f(k) \cdot \mathrm{poly}(|I|)$ for some computable function f. Not all parameterized problems are fixed-parameter tractable. Downey and Fellows [16] developed a theory of parameterized intractability by means of devising a completeness program with complexity classes. The first level of (presumable) parameterized intractability is captured by the complexity class W[1]. There is good reason to believe that problems that are hard for W[1] are not fixed-parameter tractable. To show this stronger form of hardness, a reduction concept is needed. A *parameterized reduction* reduces a problem instance (I, k) in $f(k) \cdot \mathrm{poly}(|I|)$ time to an instance (I', k') such that (I, k) is a yes-instance if and only if (I', k') is a yes-instance and k' only depends on k but not on $|I|$. If for a given parameterized problem L there is a parameterized problem L' such that L' is W[1]-hard and there is a parameterized reduction from L' to L, then L is also W[1]-hard. A problem is W[1]-complete if it is W[1]-hard and also contained in W[1]. See Chen and Meng [11] for a recent survey on parameterized hardness.

The *color-coding* technique yields randomized fixed-parameter algorithms; it was introduced for special cases of the SUBGRAPH ISOMORPHISM problem by Alon et al. [2]. The main idea is to randomly color the vertices of the graph, and then to solve the corresponding problem under the assumption that the subgraph that is searched for obtains a *colorful* coloring, that is, all of the vertices of the subgraph have pairwise different colors. This assumption often leads to an easier problem.

The whole procedure of coloring and then solving the subsequent problem on the colored graph is repeated until the subgraph that is searched for has obtained a colorful coloring at least once with high probability. We say that a randomized algorithm solves a problem with *error probability* ϵ if the probability that it fails to return the correct answer is at most ϵ. Note that the randomized algorithms in this work do not make false positive errors, that is, if the algorithm returns that a graph has an occurrence of a motif, then such an occurrence does indeed exist.

3 Searching for Connected Motifs

In this section, we provide and analyze fixed-parameter algorithms for List-Colored Graph Motif and the parameters motif size $|M|$ and solution size $|S|$. Note that, since $|S| \leq |M|$, fixed-parameter tractability with respect to $|S|$ implies fixed-parameter tractability with respect to $|M|$. On the contrary, the inverse is not true since $|S|$ can be *much* smaller than $|M|$. We provide fixed-parameter algorithms for both parameters because for the parameter $|M|$ a better worst-case running time bound can be achieved. Furthermore, we study the parameterized complexity of List-Colored Graph Motif for the dual parameter $n - |M|$.

3.1 Parameter Motif Size

We present a color-coding algorithm that partially resembles the algorithm for Graph Motif by Fellows et al. [18]. The idea is to randomly assign to each vertex one out of $|M|$ *labels*. Then, in case all vertices of an occurrence have received pairwise different labels (we call this event a *good labelling*), we can use dynamic programming to find this occurrence.

Theorem 1: List-colored Graph Motif can be solved with error probability ϵ in $O(|\ln(\epsilon)| \cdot 10.88^{|M|} \cdot m)$ time.

Proof: Without loss of generality, we can assume that M is colorful. Otherwise, we can transform M and G as follows: For each color c that occurs $\mathrm{occ}(c)$ times, we add $\mathrm{occ}(c)$ new colors to M and completely remove c from G. Furthermore, for every vertex v in G with $c \in \mathrm{col}(v)$, we remove c from $\mathrm{col}(v)$ and add the $\mathrm{occ}(c)$ new colors to $\mathrm{col}(v)$. Let M' and G' be the thus modified motif and graph, respectively. We now solve the problem of finding an occurrence of M' in G'. Each such occurrence clearly corresponds to an occurrence of M in G.

Let $L = \{l_1, l_2, \ldots, l_{|M|}\}$ denote a set of $|M|$ distinct labels. We randomly assign the labels of L to the vertices of the graph and solve the problem of finding an occurrence of the motif M under the assumption that all vertices of the occurrence have received a different label. The problem of finding a colorful occurrence of M that has the labels of L is solved by dynamic programming. First, we extend our notion of occurrence. Let $F \subseteq (L \cup M)$ be a set that contains labels as well as colors. An *occurrence of* F is defined as a set of vertices S such that the vertices of S have exactly the labels of $F \cap L$, and there is an injection $f : S \to F \cap M$ such that $f(v) \in \mathrm{col}(v)$ for each vertex v. The idea is that with the set F we store both the set of labels $L \cap F$ that we have "used" so far, and the colors that the vertices with the labels in $L \cap F$ are assigned. This way, we can find for each combination of sets $M' \subseteq M$ and $L' \subseteq L$, $|L'| = |M'|$, an occurrence of M' that has the labels of L'.

We use two dynamic programming tables D and T. The table D has entries for each vertex, and sets $F \subseteq L \cup M$. The aim of the dynamic programming procedure is to compute the values of D such that $D_v(F) = 0$ if there exists an occurrence of F that contains v, and $D_v(F) > 0$, otherwise. The table T has entries for each vertex v, each color $c \in \mathrm{col}(v)$ and sets $F \subseteq L \cup M$. The aim is to compute the values of T such that $T_{v,c}(F) = 0$ if there is an occurrence of $F \uplus \{\mathrm{label}(v), c\}$ in which v receives the color c, and $T_{v,c}(F) > 0$, otherwise. Clearly, for each v, we have to create $T_{v,c}(F)$, only for c with $c \in \mathrm{col}(v)$ and for F such that $c \notin F$ and $v \notin F$.

We initialize the table T with $T_{v,c}(\emptyset) = 0$. In the recursion, we fill in the values for increasing sizes of F for both T and D. First, we use table T to calculate values of D:

$$D_v(F) = \min_{c \in \mathrm{col}(v)} \{T_{v,c}(F \setminus \{\mathrm{label}(v), c\}), 1\}.$$

The correctness of this recurrence follows from the definition of the table entries. Then, we calculate the value for $T_{v,c}(F)$ branching into two cases. The first case is that there is a neighbor u of v such that there is an occurrence of F that contains u. The second case is that F can be partitioned into two sets F' and $F \setminus F'$ such that there is an occurrence of $F' \cup \{\mathrm{label}(v), c\}$ and of $(F \setminus F') \cup \{\mathrm{label}(v), c\}$ such that in both occurrences v is contained and has color c. The "score" for these occurrences can be found in the table T. The recurrence reads as follows:

$$T_{v,c}(F) = \min_{\substack{u \in N(v), \\ F' \subset F}} \left\{ \begin{array}{l} D_u(F), \\ T_{v,c}(F') + T_{v,c}(F \setminus F') \end{array} \right\}.$$

If there is a $v \in V$ such that $D_v(L \cup M) = 0$, then there is an occurrence of $L \cup M$ in G. A maximum-cardinality occurrence can be found by finding a vertex v and a maximum-cardinality set F such that $D_v(F) = 0$. The vertex set that corresponds to the occurrence can be computed by doing a simple traceback.

For the running time consider the following. Clearly, $|L \cup M| = 2|M|$. The recurrence for table D and the first part of the recurrence for table T can be computed in $O(2^{2|M|} \cdot |M| \cdot m)$ time overall. For the second part of the recurrence of table T, Björklund et al. [5] showed that recurrences of this type can be solved in $2^x \cdot \mathrm{poly}(x)$ time, when the base set over which the table is defined has size x. Here, this base set is $L \cup M$ and it has size $2 \cdot |M|$. This results in a running time of $2^{2 \cdot |M|} \cdot \mathrm{poly}(2|M|) = 4^{|M|} \cdot \mathrm{poly}(|M|)$ for the dynamic programming procedure for each table $T_{v,c}$, and thus to a

running time of $4^{|M|} \cdot \text{poly}(|M|) \cdot n$ for all such tables. For the random labelling, each vertex of V is labelled with one of $|M|$ labels under uniform distribution. Then, the probability of a good labelling is $|M|!/|M|^{|M|} > e^{-|M|}$. Therefore, the number of trials needed to obtain a good labelling with probability $1 - \epsilon$ is $O(|\ln(\epsilon)| \cdot e^{|M|})$. The total running time thus amounts to $O(|\ln(\epsilon)| \cdot e^{|M|} \cdot 4^{|M|} \cdot \text{poly}(|M|) \cdot m) = O(|\ln(\epsilon)| \cdot 10.88^{|M|} \cdot m)$. □

3.2 Parameter Solution Size

In this section, we present an algorithm for the parameter solution size $|S|$, applying an idea of Dondi et al. [14] to our algorithm from Section 3.1. The approach can be roughly described as follows. Let $S \subseteq V$ be a solution of LIST-COLORED GRAPH MOTIF, and let $M' \subseteq M$ be a submotif such that there is an injection f from S to M' with $f(v) \in \text{col}(v)$ for each $v \in S$. Clearly, to distinguish the vertices of S using color-coding, one only needs to use $|S|$ many labels. However, we do not know in advance what colors the color set M' comprises. Hence, the idea is to use a further color-coding step this time mapping the colors of M to the colors of a newly created set $M_{|S|}$ of size $|S|$ in order to obtain an equivalent instance that has a motif of size $|S|$. The problem of finding an occurrence of $M_{|S|}$ is fixed-parameter tractable with respect to the parameter $|S|$ since we can apply our algorithm from Section 3.1. This extends the fixed-parameter tractability result of Dondi et al. [14] for GRAPH MOTIF with respect to the parameter $|S|$ to LIST-COLORED GRAPH MOTIF. In the following, we present the details of the algorithm and bound its running time.

Theorem 2: LIST-COLORED GRAPH MOTIF can be solved in $O(|\ln(\epsilon)| \cdot 29.6^{|S|} \cdot |S| \cdot m)$ time.

Proof: The algorithm proceeds as follows. Starting with $k = 1$, it finds an occurrence of size k if such an occurrence exists. The value k will be incremented by one as long as a solution has been found. If for some value of k the algorithm fails to find a solution, then with high probability no size-k solution exists and the algorithm reports the solution of size $k - 1$ that was found previously. We now describe in detail how the algorithm works for a fixed value of k. First, create a new set of k colors $M_k := \{c_1, \ldots, c_k\}$. Then, construct a mapping $\phi : M \to M_k$ by mapping each color $c \in M$ uniformly at random to a color in M_k. Create a new graph G' from G as follows. For each vertex v and each $c \in \text{col}(v)$, remove c from $\text{col}(v)$ and add $\phi(c)$ to $\text{col}(v)$. We now use the algorithm from Section 3.1 to find an occurrence of M_k in G'. If no occurrence was found, then repeat the procedure above without changing k until either an occurrence was found, or we can, with sufficiently low error probability, conclude that G contains no occurrence of a size-k subset of M.

First, we show that if there is a size-k set $M' \subseteq M$ such that there is an occurrence S of M' in G, then with probability at least e^{-k} we create an instance that has an occurrence of M_k in G': Since each color of M' is mapped uniformly at random to one of the colors of M_k, the probability that all colors of M' are mapped to pairwise different colors of M_k is at least $k!/k^k > e^{-k}$. In this case, S is an occurrence of M_k in G'.

Second, we show that if the algorithm finds an occurrence S of M_k in G', then there is also a size-k set $M' \subseteq M$ such that S is an occurrence of M' in G. Let S be an occurrence of M_k in G', and let f be an injection from S to M_k (which must exist since S is an occurrence of M_k). Since f is an injection, $f(v) \neq f(u)$ for each pair of vertices $u, v \in S$, $u \neq v$. Hence, by choosing for each vertex v of S an arbitrary color c such that $\phi(c) = f(v)$, we obtain a set M' of k pairwise different colors such that S is an occurrence of M' in G.

We now bound the running time of the algorithm for some fixed k. Suppose there is a size-k set $M' \subseteq M$ such that there is an occurrence S of M' in G. The probability, that each color of M' was mapped to a different color in M_k is at least e^{-k}. The problem of finding an occurrence of M_k in G' can be solved with constant error probability in $O(10.88^k \cdot m)$ time by using the algorithm from Section 3.1. Hence, the probability that this algorithm finds an occurrence of M_k in G' (and consequently an occurrence of some size-k subset $M' \subseteq M$ in G) is at least $O(e^{-k})$. Repeating the procedure of coloring M and then applying the algorithm from Section 3.1 $O(e^k)$ times, a size-k occurrence of some $M' \subseteq M$, if such an occurrence exists, is found with constant probability. Hence, for fixed ϵ we can solve the problem of finding with constant error probability an occurrence of some size-k subset of M in $O(|\ln(\epsilon)| \cdot e^k \cdot 10.88^k \cdot m) = O(|\ln(\epsilon)| \cdot 29.6^k \cdot m)$ time. If no such occurrence has been found, we can conclude that with probability at least $1 - \epsilon$ no such occurrence exists.

In the overall algorithm loop, we abort as soon as $k = |S| + 1$, since by definition of the solutions size $|S|$, no occurrence of a size-$(|S| + 1)$ subset of M exists. The overall running time bound follows. □

3.3 Dual Parameterization

We study the parameterized complexity of LIST-COLORED GRAPH MOTIF for the so-called *dual* parameter $n - |M|$. At first, this parameterization appears to be uninteresting since the motif is very small compared to the network. However, in some applications, there are many vertices of the input graph that can be removed by a simple data reduction since their color-lists do not contain any color of the motif. After this data reduction, the remaining graph has modest size, and furthermore often contains several connected components (this observation was also made by Bruckner et al. [9]). For many of these components, the motif is relatively large compared to the order of the connected component. Then, only few vertices may be "removed" from this component to obtain the motif. Unfortunately, in general the LIST-COLORED GRAPH MOTIF problem becomes W[1]-hard as we show later

in this section. However, for the simple case, when the graph is vertex-colored instead of list-colored and the motif is colorful, we obtain fixed-parameter tractability. The idea of the corresponding search tree algorithm is to branch on vertices in G that have the same color. Each occurrence of the motif contains at most one of these two vertices since the motif is colorful. In the following, we describe this algorithm in detail.

Theorem 3: For colorful motifs and vertex-colored graphs LIST-COLORED GRAPH MOTIF can be solved in $O(2^{n-|M|} \cdot m)$ time.

Proof: Given a vertex-colored graph $G = (V, E)$ and a colorful motif M, the algorithm proceeds as follows. Initially, set $d := n - |M|$. Clearly, we can assume that every color of M appears in G, otherwise we can simply remove this color from M. Furthermore, let M' be a maximum-cardinality subset of M such that there is an occurrence of M' in G. For $d > 1$, the graph must contain two vertices u and v such that $\mathrm{col}(v) = \mathrm{col}(u)$. At most one of these vertices belongs to an occurrence of M'. Accordingly, the algorithm recursively finds occurrences of M' in $G[V \setminus \{u\}]$ and in $G[V \setminus \{v\}]$. In each recursive branch, set $d := d-1$. In case $d = 0$, the graph G contains exactly the colors of M. Then, the largest occurrence of a subset of M is simply the largest connected component of G. There is at least one search tree leaf in which G has a connected component whose colors are M'. Hence, the overall solution is the largest occurrence that was found over all recursive calls in the search tree.

As to the running time, the search tree has size $O(2^d)$ since it has depth d and branches into two cases at each search tree node. Furthermore, the operations at each search tree node can be performed in $O(m)$ time. □

Unfortunately, Theorem 3 does not carry over to the general LIST-COLORED GRAPH MOTIF problem. On the contrary, we show that the problem becomes W[1]-hard for vertex-colored graphs and motifs that consist of two colors, and also for colorful motifs when the input graph is list-colored.

Theorem 4: LIST-COLORED GRAPH MOTIF is W[1]-hard with respect to the parameter $n - |M|$ even if the input graph G is vertex-colored and the motif M consists of two colors.

Proof: We reduce from the W[1]-complete INDEPENDENT SET [16] problem:

> *Input:* An undirected graph G and a nonnegative integer k.
> *Question:* Is there a size-k vertex set S such that $G[S]$ has no edges?

Given an instance $(G = (V, E), k)$ of INDEPENDENT SET, we build an instance of LIST-COLORED GRAPH MOTIF as follows. The motif M is a multiset over two colors b and w such that M contains the color b exactly $|V|-k$ times and the color w exactly $|E| + 1$ times. Each vertex $v \in V$ is colored with color b. Furthermore, each edge $\{u, v\} \in E$ is replaced by a path of length two, that is, we remove $\{u, v\}$ from G, add a new vertex $e_{\{u,v\}}$ to G and insert edges between u and $e_{\{u,v\}}$ and v and $e_{\{u,v\}}$. Each new vertex receives the color w. Finally, we add one further vertex e^* with color w and add edges between e^* and every vertex that has color b. Let G' denote the resulting vertex-colored graph. The construction can be clearly performed in polynomial time. Note that $n - |M| = k$. We complete the proof by showing that

> G has a size-k independent set $\Leftrightarrow$ G' has an occurrence of M.

"$\Rightarrow$" Let S be a size-k independent set in G. Consider the graph G'' that is obtained from G' by removing S. We show that G'' is an occurrence of M. Consider the colors of G''. Since all k vertices that have been removed from G' have color b there are $|V| - k$ vertices that have color b and $|E| + 1$ vertices that have color w in G''. It remains to show that G'' is connected. Since e^* is adjacent to all vertices of V, the subgraph that is induced by $V \cup \{e^*\}$ is connected. Furthermore, every other vertex is adjacent to at least one vertex of V. Suppose that this is not the case, then for some vertex $e_{\{u,v\}}$ both u and v are in S. This contradicts the fact that S is an independent set in G.

"$\Leftarrow$" Let S be a vertex set such that removing S from G' results in an occurrence of M, that is, a graph that is connected and has the colors of M. By construction of G' and M, S has size k and contains only vertices that have color b. Hence, these vertices correspond to vertices of G. We show that S is an independent set in G. Suppose that this is not the case, then there must be two vertices $u, v \in S$ such that $\{u, v\} \in E$. Then, however, both neighbors of $e_{\{u,v\}}$ in G' are in S. This contradicts the fact that removing S from G' results in a connected graph. □

By slightly modifying the construction in the proof of Theorem 4, we can also transfer this result to the case that the motif is colorful. The only difference is that instead of using two colors b and w, we use two sets of colors $B = \{b_1, \ldots b_{n-k}\}$ and $W = \{w_1, \ldots w_{m+1}\}$. Every color that was colored by b now receives the color list B and every color that was colored by w receives the color list W. The motif is $B \cup M$. The correctness proof works analogously.

Theorem 5: LIST-COLORED GRAPH MOTIF is W[1]-hard with respect to the parameter $n-|M|$ even if M is colorful.

Theorems 4 and 5 exclude any hope for fixed-parameter algorithms for LIST-COLORED GRAPH MOTIF parameterized by the dual parameter $n-|M|$. However, as we show in Section 4.2, there are some cases in which even a brute force algorithm for guessing the set of vertices to delete is faster than the color-coding fixed-parameter algorithms for parameters $|M|$ and $|S|$, respectively. Hence, it seems worthwhile to find further special cases in which LIST-COLORED GRAPH MOTIF is fixed-parameter tractable with respect to $n-|M|$ or to study $n-|M|$ in combination with other parameters.

4 APPLICATION TO QUERYING OF PROTEIN-INTERACTION NETWORKS

We applied our algorithms for LIST-COLORED GRAPH MOTIF to topology-free querying of protein-interaction

networks. As described in the introduction, the input of the graph motif instance here is a colorful motif and the vertices in G are list-colored. Our program is written in the C++ programming language, uses the Boost Graph Library[3], and consists of roughly 1000 lines of code. The source code is publicly available.[4]

4.1 Implementation Details

This section describes some details that distinguish the implemented algorithms from those specified in Section 3. Given a graph $G = (V, E)$ and a colorful motif M, our implemented algorithms not only find a maximum-cardinality subset $S \subseteq V$ such that $G[S]$ is an occurrence of M, but also compute S such that the weight of a spanning tree of $G[S]$ is maximized. Our implementation provides three algorithms: the first is a simple (randomized) brute force approach. This algorithm is not used on its own, but as a subroutine of the other two algorithms. The second algorithm solves LIST-COLORED GRAPH MOTIF with the parameter motif size $|M|$ as described in Section 3.1. The third algorithm solves LIST-COLORED GRAPH MOTIF with the parameter solution size $|S|$ as described in Section 3.2. The algorithm with parameter $|M|$ and the algorithm with parameter $|S|$ resort to the brute force approach if it is expected to outperform the dynamic programming routines described in Section 3.1. This is, for example, the case if the dual parameter (Section 3.3) is small.

4.1.1 Randomized Brute Force

Given a LIST-COLORED GRAPH MOTIF instance comprising a graph $G = (V, E)$, a colorful motif M, and a natural number k, this algorithm proceeds as follows: it chooses a size-k set $S \subseteq V$ uniformly at random and checks whether $G[S]$ is an occurrence of a subset of M. This verification step is carried out by computing a maximum-cardinality matching in the bipartite graph $H = (S \uplus M, F)$, where an edge $\{v, c\} \in F$ with $v \in S$ and $c \in M$ exists if and only if $c \in \mathrm{col}(v)$. If all vertices in S are matched and $G[S]$ is connected, then $G[S]$ is an occurrence of a size-k subset of M. The process of randomly choosing subsets of G is repeated a sufficient number of times so that an existing occurrence of a size-k subset of M is found with high probability. More precisely, the probability that a size-k occurrence is chosen is at least $1/\binom{|V|}{k}$. Hence, we repeat the procedure of guessing and verifying a solution $O(|\ln(\epsilon)| \cdot \binom{|V|}{k})$ times to obtain an error probability of at most ϵ. For each motif occurrence found in the progress, its maximum-weight spanning tree is computed. The occurrence with the maximum-weight spanning tree among all size-k occurrences is reported. Since we have to repeat the procedure $O(|\ln(\epsilon)| \cdot \binom{|V|}{k})$ times, the algorithm works fast if k is small or close to $|V|$. In the latter case, the dual parameter (Section 3.3) is small. However, Randomized Brute Force is not a fixed-parameter algorithm with respect to the parameter k or the parameter $|V| - k$.

4.1.2 Parameter Motif Size

In contrast to the algorithm described in Section 3.1, the implemented algorithm does not employ the (theoretical) result by Björklund et al. [5] to quickly evaluate the given recurrences. Instead, they are straightforwardly computed by dynamic programming. However, we implemented heuristics to reduce the running time of the algorithm.

Overall Strategy

Each connected component of the input graph G is processed independently. For each connected component, the Randomized Brute Force procedure described in Section 4.1.1 is applied if it is expected to outperform the color-coding algorithm from Section 3.1. Otherwise, the color-coding algorithm is invoked. Next, we describe the implementation details of the color-coding algorithm.

Increasing the Success Probability of Color-Coding

The running time depends to a large extent on the number of trials needed to achieve a good labelling with sufficiently high probability. We implemented two approaches to increase the probability of a motif occurrence to receive a good labelling. This, in turn, reduces the number of trials needed to achieve a sufficiently low error probability. The two approaches are described in the following two paragraphs.

Separating Color Sets: This approach to increase the probability of a motif occurrence to receive a good labelling is due to Bruckner et al. [9]. Assume that there is a *separating* color subset C such that the color list of each vertex either contains *no* colors from C or *exclusively* colors from C. In this case, we can improve on standard color-coding. Let V_1 be the vertices that contain only colors from C and let V_2 be the vertices that contain no colors from C. If the given motif contains k_1 colors from C and k_2 colors not from C, then the occurrence of the motif must contain k_1 vertices from V_1 and k_2 vertices from V_2. Thus, we may draw the labels for V_1 and V_2 from disjoint label sets L_1 and L_2, respectively. The probability for a good labelling is the probability that the vertices in V_1 and V_2 receive a good labelling. This results in a good labelling of vertices with a probability of $(k_1!/k_1^{k_1}) \cdot (k_2!/k_2^{k_2})$ instead of $(k_1 + k_2)!/(k_1 + k_2)^{k_1+k_2}$. Separating color subsets, if they exist, can be computed in $O(k \cdot m)$ time by finding connected components in an auxiliary graph [9].

Injective Color-Coding: Assume that C is a separating color set, as described above, such that the sought colorful motif M contains k colors from C. Moreover, let V_C be the vertices that contain only colors of C. Observe that the probability that V_C receives a good labelling is $k!/k^k$ while the probability of guessing a vertex subset of V_C that is part of a motif-occurrence is $1/\binom{|V_C|}{k}$. As a result, if the latter probability is higher

3. http://www.boost.org/
4. http://theinf1.informatik.uni-jena.de/graph-motif

than the former, then we avoid the standard color-coding technique. Instead, we choose a random subset of V_C and label its vertices injectively.

Our experiments (see Section 4.2) showed that the combination of these two heuristics reduces the number of color-coding trials tremendously.

4.1.3 Parameter Solution Size

As described in Section 3.2, the algorithm works by iterating over all possible solution sizes from $k := 1$ to $|S| + 1$. Color-coding is applied repeatedly to obtain a colorful motif $M' \subseteq M$ with $|M'| = k$. In each repetition, the algorithm for the parameter motif size (Section 4.1.2) is applied to M' and G (with new color lists, as described in Section 3.2). We use the following two approaches to heuristically improve the running time of this algorithm:

Injective Color-Coding: Similar to Section 4.1.2, Injective Color-Coding may be applied to increase the probability that a size-k subset of M with an occurrence in G receives a good coloring. This applies if $k!/k^k$ is less than $1/\binom{|M|}{k}$. In this case, we reduce the number of trials needed to obtain good colorings for the subsets of M by injectively coloring a randomly chosen size-k subset of M.

Lower Bounds for Solution Size: The algorithm for the parameter solution size checks whether size-k subsets of M have occurrences in G, iterating over increasing values of k and starting with $k = 1$. A lower bound on the solution size allows us to skip many of these iterations by not increasing k by only one, but setting it to the currently best known lower bound. Such a lower bound on the solution size can be obtained using Random Occurrence Guessing (see Section 4.1.1). We also use Randomized Brute Force to find size-$(|M| - k + 1)$ occurrences in G if this approach is expected to outperform the color-coding algorithm for finding size-k occurrences.

As for the algorithm for parameter $|M|$, these two heuristics reduce the number of color-coding trials needed before a solution is found (see Section 4.2). However, the overall number of trials needed is usually higher than for the algorithm with parameter $|M|$. The reason is that we have to perform color-coding twice: once for the motif colors and once for the vertex labels.

4.2 Experiments

Data Acquisition

We tested our algorithms on three weighted protein-interaction networks (the weights denote interaction probabilities) from yeast (5430 proteins, 39936 interactions), fly (6650 proteins, 21275 interactions), and human (7915 proteins, 28972 interactions) that were assembled by Bruckner et al. [9]; the queries were complexes from the same three species. Each query protein was identified with a unique color, the proteins of the networks received the colors of all query proteins whose sequence similarity to the network protein exceeded a predefined threshold. This threshold was set to a BLAST score of 10^{-7}. We considered the following four combinations of network and complexes (for which pairwise BLAST scores were available to us): we queried yeast complexes in the human network and in the fly network, human complexes in the yeast network, and fly complexes in the yeast network. Table 1 shows the total number of instances (grouped into categories of small, medium, and large motif size).

Computational Setting

The experiments have been executed on a standard desktop PC with 2.2 GHz Athlon64 CPU and 1 GiB of RAM. For each network-motif pair, we aborted execution after 10 minutes or if the memory limit of 900 MiB was exceeded.

For each pair of complex (equivalently, motif) and network, we computed the largest subset of the motif that has an occurrence in the network. Of these, we output the occurrence with the spanning tree of maximum weight. This implies that we cannot stop the color-coding process after a motif occurrence has been found. Instead, we have to execute all color-coding trials in order to find the maximum-weight motif occurrence with the given error probability.

Experimental results

Table 1 summarizes the results. The average running time of the solved instances is lower for the algorithm with parameter $|S|$ (Section 4.1.3) than for the algorithm with parameter motif size $|M|$ (Section 4.1.2). However, the algorithm with parameter $|M|$ is able to solve more instances within the running time limit of 10 minutes. As one would expect, these are the instances where the motif occurrences were large. This is also the reason why the average size of the occurrences found by the algorithm with parameter $|S|$ is smaller. It is notable that, while we set an upper limit of 10 minutes to solve an instance, all instances solved within this time limit were solved in a few seconds. Most instances could be solved within milliseconds. We conclude that usually the algorithm with parameter $|M|$ should be applied instead of the algorithm with parameter $|S|$, since it was able to solve more instances and is outperformed by the algorithm for parameter $|S|$ only for "easier" instances.

The effect of the heuristics for reducing the number of color-coding trials (see Sections 4.1.2 and 4.1.3) is also shown in Table 1. For both algorithms, there is a difference of several orders of magnitude between the number of color-coding trials that are needed with and without heuristics, and this difference is especially pronounced for the more difficult instances with large motifs. Furthermore, we examined how the inclusion of the Randomized Brute Force algorithm helps in reducing the overall running time for both algorithms. The running times of the algorithms without the Randomized Brute Force subprocedure are shown in Table 1 c) and d), respectively. For both algorithms there is a decrease in the number of solved instances and an increase in the average running times, demonstrating the usefulness of Randomized Brute Force for some instances. Furthermore,

	# instances		avg. size		time (secs)		avg. # coloring trials		# instances with occ. size			
$\|M\|$	solved	unsolved	$\|M\|$	$\|S\|$	avg.	max.	standard	improved	100%	99%-50%	49%-25%	25%-0%
1–7	1089	0	2.20	1.27	0.00	0.07	36.43	1.07	539	345	169	36
8–14	47	2	10.55	4.09	2.62	59.08	$9.7 \cdot 10^4$	22.23	0	15	12	20
≥ 15	18	3	21.28	5.33	0.43	7.59	$4.1 \cdot 10^{12}$	8.28	0	4	2	12

(a) Results for the algorithm with parameter $|M|$ (Section 4.1.2).

	# instances		avg. size		time (secs)		avg. # coloring trials		# instances with occ. size			
$\|M\|$	solved	unsolved	$\|M\|$	$\|S\|$	avg.	max.	standard	improved	100%	99%-50%	49%-25%	25%-0%
1–7	1089	0	2.20	1.27	0.00	0.08	416.35	0.82	539	345	169	36
8–14	43	6	10.55	3.63	1.50	55.05	$2.8 \cdot 10^8$	65.44	0	11	12	20
≥ 15	14	7	21.28	3.43	0.02	0.07	$1.1 \cdot 10^{13}$	41.72	0	0	2	12

(b) Results for the algorithm with parameter $|S|$ (Section 4.1.3)

	# instances		avg. size		time (secs)	
$\|M\|$	solved	unsolved	$\|M\|$	$\|S\|$	avg.	max.
1–7	1089	0	2.20	1.27	0.00	0.44
8–14	42	7	10.45	3.43	8.12	113.82
≥ 15	14	7	21.71	3.43	0.10	0.72

(c) Algorithm for parameter $|M|$ without Randomized Brute Force

	# instances		avg. size		time (secs)	
$\|M\|$	solved	unsolved	$\|M\|$	$\|S\|$	avg.	max.
1–7	1089	0	2.20	1.27	0.00	0.60
8–14	41	8	10.39	3.27	6.71	159.17
≥ 15	14	7	21.71	3.43	0.18	1.75

(d) Algorithm for parameter $|S|$ without Randomized Brute Force

TABLE 1: Experimental results. Unsolved instances needed either more than 10 minutes of time or more than 900 MiB of space. All values shown are for the set of solved instances only. All instances were processed with a maximum error probability of 0.1%.

since the average occurrence size of the solved instances is smaller when Randomized Brute Force is not used, we conclude that Randomized Brute Force is useful when $|S|$ is relatively large.

4.3 Related implementations

Bruckner et al. [8, 9] developed a web-service tool (TORQUE) to solve an extension of our problem allowing for insertions of vertices that are not part of the motif. They combine several algorithms: a color-coding procedure for small motifs, an Integer Linear Programming formulation for large motifs, and a shortest-path based heuristic. The main difference of our implementation compared to TORQUE is that TORQUE allows for insertion of additional/uncolored vertices in order to establish the connectivity of the solution set. That is, TORQUE reports solutions that contain uncolored vertices that connect different connected components of colored vertices if this results in a larger overall occurrence size. Furthermore, the number of deletions, that is, the number of motif colors that are *not* in an occurrence, is limited for TORQUE (for example, for motifs/queries of size 10, the occurrence has to contain at least 6 vertices that are not colored). In summary, the solutions of TORQUE are usually larger than our solutions, and not all of the uncolored vertices of the network can be removed by data reduction, since some of them might be needed to connect colored vertices. Hence, we solve a different problem, and our running times do not compare directly to the ones of Bruckner et al. [9]. Blin et al. [6] provide a Cytoscape plug-in (based on a Linear Pseudo-Boolean optimization solver) for LIST-COLORED GRAPH MOTIF with insertions. Again, this makes a comparison of running times difficult, since our algorithm solves a more restricted problem.

Our color-coding algorithm can handle larger motifs than the color-coding algorithm of Bruckner et al. [9] who use Integer Linear Programming to handle queries/motifs of size > 10. Furthermore, our algorithm almost always terminates within few seconds which is not the case for the algorithm of Bruckner et al. [9]. This encourages both the application of our algorithm in case insertions are explicitly forbidden and the extension of our algorithm to the more general problem that allows for insertions. One advantage of our approach is that the overall number of solutions is much larger than for TORQUE since we always report the largest occurrence that was found, that is, we allow an arbitrary number of deletions. Hence, it can be applied in case TORQUE does not yield a solution.

4.4 Functional Enrichment of Predicted Complexes

We examine the quality of the solutions that were found by our algorithm, by assessing the enrichment of functional terms in the protein sets of the solutions. For each solution, we retrieved functional annotation terms from the SGD database [28] (for the yeast network), the GOA database [3] (for the human network), or FlyBase [31] (for the fly network). Then, we used the GO::TermFinder tool [7] to find functional annotation terms that have a statistically significant overrepresentation compared to random protein sets. This is done by computing for each functional annotation term the p-value of its abundance in the solution under the hypothesis that the solution does not show an overrepresentation of this functional annotation term. The reported p-values are corrected for

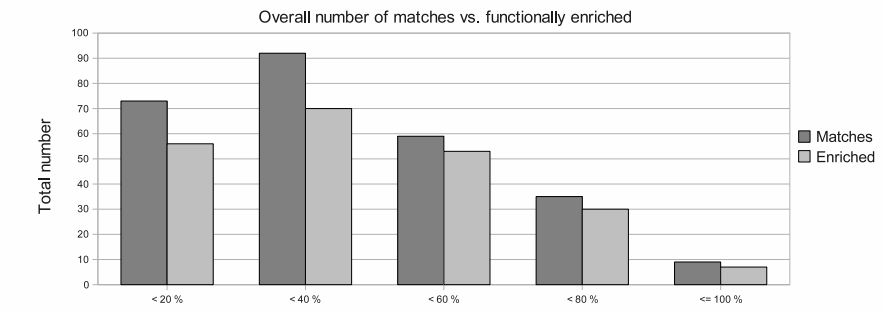


Fig. 1: Comparison of the number of solutions with the number of solutions that showed a significant enrichment of at least one functional annotation term. Along the x-axis, the solutions are grouped into five categories, according to the value of solution size/query size.

multiple hypotheses testing using the Bonferroni method and the threshold for considering an enrichment as significant was set to $p < 0.05$. Since the solutions should be complexes, they are expected to have a common function. Hence, the percentage of solutions that have a common function is a measure of the solution quality.

First, we examined how the percentage of functionally enriched solutions correlates with solution sizes. We found that of the solutions with size two or three, which make up roughly 85% of the solutions, more than 78% have a significant enrichment of at least one functional annotation term. Of the solutions of size at least four, 100% had a significant enrichment of at least one functional annotation term. We therefore examined how the relation between query size and solution size influences the solution quality. Our results are shown in Figure 1. For solutions whose size is more than 40% of the query, the ratio of enriched vs non-enriched is roughly the same. For solutions whose size is less than 40% of the query, the percentage of enriched solutions drops. We thus conjecture that the number of allowed deletions should be set to roughly 50% of the query size. For some instances, this is more than the number of deletions that are allowed by TORQUE, and possibly the percentage of instances that have a solution in TORQUE could be increased without a drop-off in solution quality by allowing more deletions. However, our results also show that by excluding insertions, the number of vertices in the solution is often very small. Hence, a limited number of insertions should be permitted. So far, however, the precise number of insertions that should be allowed seems to be unexplored.

5 On Finding More Robust Motifs

Lacroix et al. [24] motivated the study of (variants) of the Graph Motif problem by considerations comparing "topological motifs" with "functional motifs". The Graph Motif problem only poses a minimal demand on the motif topology by requiring connectedness. The question arises what happens if we ask for somewhat "more robust" motifs, replacing the connectedness demand by standard graph-theoretic demands for biconnectivity, bridge-connectivity, and the like. Surprisingly, as we will show in this section, these seemingly small steps towards topologically more constrained motifs already lead to W[1]-hardness results, destroying the hope for fixed-parameter algorithms for these Graph Motif variants. Indeed, we will prove even stronger results, probably of independent interest, by showing that the problems of deciding on the existence of fixed-size biconnected or bridge-connected subgraphs, parameterized by the size of the subgraphs, are W[1]-hard. Moreover, we extend these results to higher connectivity demands. Furthermore, we answer an open question of Dondi et al. [13] by showing that the parameter "number of connected components" in a graph motif leads to a W[1]-hard problem.

5.1 Biconnected Subgraphs of Size Exactly k

Originally, the Graph Motif problem was suggested, because it imposes the least possible restriction on the topology of the occurrence of the motif [24]. One way of extending the Graph Motif problem in this spirit is to search for *biconnected* occurrences instead of connected occurrences of the motif. Recall that a graph is biconnected if it has no cut-vertex. For example in the scenario of protein-interaction network querying, demanding biconnectivity could be used to demand occurrences with higher interaction probabilities, without restricting the actual topology of the occurrence too much. The decision version of the problem can be formulated as follows:

> Biconnected Graph Motif
> *Input:* A vertex-colored undirected graph $G = (V, E)$ and a multiset of colors M.
> *Question:* Does there exist an $S \subseteq V$ such that the induced subgraph $G[S]$ is biconnected and there is a bijection between the colors of the vertices in S and M?

We will show that Biconnected Graph Motif is W[1]-complete when parameterized by the size of the motif M. In fact, we prove an even stronger result. Consider the special case that M contains only one color c, $|M| = k$, and all vertices in G have color c. Then, the resulting problem is to find a biconnected subgraph of size *exactly* k:

Fig. 2: An example of the transformation of a CLIQUE instance with $k = 3$ into a BICONNECTED SUBGRAPH instance with $k' = 15$. White vertices in G' belong to V_1, black vertices to V_2.

BICONNECTED SUBGRAPH:
Input: An undirected graph $G = (V, E)$ and a nonnegative integer k.
Question: Does there exist an $S \subseteq V$ of size k such that the induced subgraph $G[S]$ is biconnected?

In the following, when talking about the *order* of a subgraph, we always refer to the number of vertices it contains. Note that looking for a biconnected subgraph of order *at least* k is solvable in polynomial time by removing all cut-vertices of the graph and then finding the largest component. However, restricting the order of the biconnected subgraph to exactly k makes the problem surprisingly hard. We prove the parameterized hardness by reduction from the CLIQUE problem, which is known to be W[1]-complete [16] with respect to the order of the clique searched for.

CLIQUE
Input: An undirected graph G and a nonnegative integer k.
Question: Is there a complete subgraph of order k in G?

Theorem 6: BICONNECTED SUBGRAPH is W[1]-complete with respect to the parameter k.

Proof: To show the W[1]-hardness, we give a parameterized many-one reduction from CLIQUE to BICONNECTED SUBGRAPH.

Let (G, k) be a CLIQUE instance. We construct a graph G' from G by replacing every edge e of G with a simple path p_e that has $\binom{k}{2} + 1$ internal new vertices. The vertex set of G' can be partitioned into two vertex sets V_1 and V_2, where V_1 contains the vertices that correspond to vertices of the original graph G and V_2 contains the new internal path vertices. An example of this transformation is shown in Figure 2. Note that the reduction works for arbitrary path length greater than $\binom{k}{2}$. For reasons of simplicity, we choose the path length to be $\binom{k}{2} + 1$.

We prove in the following that G has a clique of order k if and only if G' has a biconnected subgraph of order $k' = k + \binom{k}{2} \cdot (\binom{k}{2} + 1)$. If G has a clique C of order k, then the subgraph that is induced by the k vertices of C and by the vertices on the $\binom{k}{2}$ paths that were created from the $\binom{k}{2}$ clique edges of C in G has order exactly $k + \binom{k}{2} \cdot (\binom{k}{2} + 1)$. Clearly, this subgraph is also biconnected.

It remains to show that if G' has a biconnected subgraph of order $k' = k + \binom{k}{2} \cdot (\binom{k}{2} + 1)$, then G has a clique of order k. Let G' have a biconnected subgraph $G'[S]$ of order k. If S contains one vertex of a path p_e, then it must contain all vertices from p_e, because otherwise $G'[S]$ would not be biconnected. Hence, the number of vertices k' in S can be expressed as $k' = a + b \cdot (\binom{k}{2} + 1)$, where $a = |S \cap V_1|$ and b denotes the number of paths in G' that correspond to edges of G.

We show that b must be $\binom{k}{2}$. First, if $b > \binom{k}{2}$, then, since we can assume without loss of generality that $k \geq 3$, $k' > (\binom{k}{2}+1) \cdot (\binom{k}{2}+1) > k + \binom{k}{2} \cdot (\binom{k}{2}+1)$, a contradiction. Second, we consider the case that $b < \binom{k}{2}$. Since it must hold that $k + \binom{k}{2} \cdot (\binom{k}{2}+1) = a + b \cdot (\binom{k}{2}+1)$, in this case one must have that $a > \binom{k}{2}$. This means that there are more than $\binom{k}{2}$ vertices in V_1 which must form a biconnected graph by inserting less than $\binom{k}{2}$ paths from V_2. Clearly, this it not possible. Hence, we have shown that $b = \binom{k}{2}$ and thus also $a = k$.

Since $G'[S]$ contains exactly $\binom{k}{2}$ paths consisting of vertices from V_2 and each path must connect two vertices of A, all vertices of A are pairwise connected via a path of length $\binom{k}{2}$. Hence, the subgraph $G[A]$ must be a clique of order k since it contains exactly k vertices and exactly $\binom{k}{2}$ edges.

Using a characterization of W[1] by Chen et al. [12], the containment of BICONNECTED SUBGRAPH in W[1] can be shown in complete analogy to Guo et al. [21, Theorem 12]. □

5.2 Bridge-connected Motifs and Motifs of Higher Connectivity

Another way to augment the connectivity demands is to search for bridge-connected motifs. We define BRIDGE-CONNECTED SUBGRAPH in complete analogy to BICONNECTED SUBGRAPH, simply replacing the demand for biconnectivity by the demand for bridge-connectivity. Recall that a graph is bridge-connected when it has no bridge, that is, an edge $\{u, v\}$ such that every path between u and v contains $\{u, v\}$. The reduction from CLIQUE as used in the proof of Theorem 6 works also for bridge-connected subgraphs. Since W[1]-membership also follows in complete analogy to [21, Theorem 12] for bridge-connected motifs as well, we can state the following theorem.

Theorem 7: BRIDGE-CONNECTED SUBGRAPH is W[1]-complete with respect to the parameter "number of subgraph vertices".

In addition, we can generalize the hardness results to higher-connected graph motifs. To this end, consider the following problem.

p-CONNECTED SUBGRAPH:
Input: An undirected graph G and a nonnegative integer k.
Question: Does there exist an $S \subseteq V$ of size k such that the induced subgraph $G[S]$ is p-connected?

Observe that p-CONNECTED SUBGRAPH is non-trivially posed only if $p \leq k$—otherwise, the answer is clearly always "No".

Theorem 8: p-CONNECTED SUBGRAPH is W[1]-complete with respect to the parameter k.

Proof: We further extend the construction used for the proof of Theorem 6 as follows: We add a set A of $p-2$ additional vertices to $G' = (V', E')$, that is, we have $V' := V_1 \cup V_2 \cup A$. Furthermore, for every vertex $a \in A$ we have an edge from a to every vertex of $V' \setminus \{a\}$. The desired motif size is increased by $p-2$, that is, we set $k' := k + \binom{k}{2} \cdot (\binom{k}{2} + 1) + p - 2$. As a p-connected component must consist of at least p vertices, we have $p \leq k$ and, thus, the new parameter k' can be expressed as a function of k.

In the following, we prove that G contains a clique of order k if and only if there is a p-connected subgraph of order k' in G'.

Given a clique of order k in G, as argued in the proof of Theorem 6, we can find a biconnected subgraph of order $k + \binom{k}{2} \cdot (\binom{k}{2} + 1)$ that contains only vertices of $V_1 \cup V_2$. Adding the vertices of A to this subgraph obviously results in a p-connected subgraph of order k'.

Given a p-connected subgraph $G'[S]$ of order $k' := k + \binom{k}{2} \cdot (\binom{k}{2} + 1) + p - 2$, we show that the vertex set $S \cap V_1$ corresponds to vertices that form a clique in G. We start by proving that A must be a subset of S. Assume that there is an $a \in A$ with $a \notin S$. Then, in $G'[V' \setminus \{a\}]$ all vertices $v_j \in V_2$ have degree $p-1$ and, hence, cannot be part of a p-connected subgraph. In addition, since $G[V' \setminus (\{a\} \cup V_2)]$ is not p-connected, it cannot contain a p-connected subgraph. Thus, we know that $A \subseteq S$ and, hence, all motif vertices are exactly $(p-2)$-connected via vertices of A. Hence, we have to choose $k + \binom{k}{2} \cdot (\binom{k}{2} + 1)$ vertices of $V_1 \cup V_2$ that increase the connectivity by two. As argued in the proof of Theorem 6, this can only be achieved by choosing vertices of V_1 that correspond to a clique.

W[1]-membership can be shown in complete analogy to Guo et al. [21, Theorem 12], and is therefore omitted. □

In complete analogy, we obtain the following result, where p-EDGE CONNECTED SUBGRAPH is defined in a similar way as p-CONNECTED SUBGRAPH, replacing the demand for p-connectedness with the demand for p-edge connectedness.

Theorem 9: p-EDGE CONNECTED SUBGRAPH is W[1]-complete with respect to the parameter k.

5.3 Min-CC Graph Motif in Paths

We consider the following variant of GRAPH MOTIF which was proposed by Dondi et al. [13] for instances possessing no connected occurrence of the complete motif M:

> MIN-CC GRAPH MOTIF:
> *Input:* A vertex-colored undirected graph $G = (V, E)$, a multiset of colors M with $|M| = k$, and a nonnegative integer d.
> *Question:* Does there exist an $S \subseteq V$ such that $G[S]$ has at most d components, and there is a bijection between the colors of the vertices in S and M?

Answering an open question of Dondi et al. [13], MIN-CC GRAPH MOTIF is W[1]-hard when parameterized by the number of the connected components of the occurrence of the motif, even if the input graph is a path.

Theorem 10: MIN-CC GRAPH MOTIF on paths is W[1]-hard with respect to the number of components.

Proof: We apply a reduction from the PERFECT CODE problem:

> Input: An undirected graph $G = (V, E)$ and a positive integer k.
> Question: Is there a size-k subset $V' \subseteq V$ such that for every vertex $v \in V$ there is exactly one vertex in $N[v] \cap V'$?

PERFECT CODE is W[1]-complete with respect to the parameter k [10].

Given a PERFECT CODE instance $(G = (V, E), k)$, we construct a MIN-CC GRAPH MOTIF instance consisting of a path P and a motif M. It asks for the existence of a solution consisting of k connected components. The vertex set of P consists of one vertex p_v with color c_v for every $v \in V$, $n-1$ "separator vertices" with color s each, and $2n$ "end vertices" with color e each. Now, we describe the ordering of the vertices in the path P. For every vertex $v \in V$ there is a subpath containing vertices that correspond to the vertices of $N[v]$ in an arbitrary order. At both ends of every subpath we add an end vertex with color e. Finally, we connect all subpaths in an arbitrary order such that two neighboring subpaths are connected through a separator vertex with color s. See Figure 3 for an example. The motif set M consists of $2k$ times the color e and $\{c_v \mid v \in V\}$.

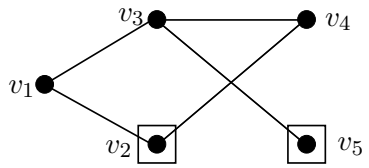
Next, we show that G has a perfect code of size k if and only if the there are k subpaths $P_1, \ldots, P_k$ such that there is a bijection between the colors of their vertices and the colors of M.

Given a perfect code $V' \subseteq V$, our motif consists of the subpaths that correspond to the vertices of V' including the end vertices on both sides. As V' is a perfect code for every $v \in V$, the color c_v appears in the subpaths exactly once and the end vertices of all subpaths give $2k$ further vertices with color e.

For the reverse direction, observe that because of the separator vertices, one connected component in the solution can contain at most two vertices with color e. Since we have $2k$ vertices of color e in the k connected subpaths of the solution, every subpath must correspond to the whole neighborhood of a vertex of V. Since every color c_v corresponding to a vertex $v \in V$ appears exactly once in the motif, the vertices that correspond to the subpaths of a solution for the MIN-CC GRAPH MOTIF instance must form a perfect code in G. □

6 CONCLUSION

LIST-COLORED GRAPH MOTIF is a natural graph-theoretic pattern matching problem with applications in the analysis of biological networks. In this work, we propose new fixed-parameter algorithms for solving LIST-COLORED



e	c_1	c_2	c_3	e	s	e	c_1	c_2	c_4	e	s	e	c_1	c_3	c_4	c_5	e	s	e	c_2	c_3	c_4	e	s	e	c_3	c_5	e
	$N[v_1]$						$N[v_2]$						$N[v_3]$							$N[v_4]$						$N[v_5]$		

Fig. 3: Example for the reduction presented in the proof of Theorem 10. The PERFECT CODE instance has vertices $v_1, \ldots, v_5$ with solution $\{v_2, v_5\}$. The first row of the table on the bottom displays the corresponding path in the MIN-CC GRAPH MOTIF instance. Herein, the color corresponding to v_i is denoted by c_i. The shaded vertices belong to a solution for MIN-CC GRAPH MOTIF.

GRAPH MOTIF. We also implemented these algorithms and tested them in the area of topology-free querying of protein-interaction networks. Our experiments showed that realistic LIST-COLORED GRAPH MOTIF instances can be solved efficiently by our implementations of our algorithms. However, we also encountered some instances where our color-coding based algorithm fails. On the positive side, some of the heuristic speed-up tricks that we used, such as the injective coloring of some color subsets and the Randomized Brute Force procedure could also prove useful for the more general problem in which insertions are allowed. Furthermore, it should be investigated how a variant of LIST-COLORED GRAPH MOTIF that finds a largest occurrence with a bounded number of *edge-insertions* compares to TORQUE [9] and to our results. This could be an appropriate way to deal with networks in which many edges are missing, without adding vertices to the solution that are not similar to any query proteins. Finally, it would be interesting to study the enumeration variant of LIST-COLORED GRAPH MOTIF that reports all occurrences of a motif. This problem is not fixed-parameter tractable with respect to $|M|$, since there could be more than $f(|M|) \cdot \mathrm{poly}(n)$ occurrences of a motif M. Consequently, other parameters need to be considered for this enumeration variant.

From the theoretical side, it would be interesting to extend the result from Section 3.3 concerning fixed-parameter tractability with respect to the parameter $n - |M|$, since only very few of the considered instances were vertex-colored instead of list-colored. One approach could be to consider additional parameters such as the size of the color lists in the list-colored graphs.

ACKNOWLEDGMENTS

We are grateful to Sharon Bruckner, Falk Hüffner, and Roded Sharan for making their protein-interaction networks, protein complex data, and BLAST score data available to us. We are furthermore grateful to Jiong Guo and Frances Rosamond for helpful comments. In particular, Jiong Guo hinted to the proof of Theorem 10. We also thank the anonymous referees for their constructive feedback.

Nadja Betzler was supported by the DFG, projects DARE, GU 1023/1, and PAWS, NI 369/10. René van Bevern was supported by the DFG, project AREG, NI 369/9. Michael R. Fellows was supported by the Australian Research Council, a main part of this work was done while he was staying in Jena as a recipient of the Humboldt Research Award of the Alexander von Humboldt Foundation, Bonn, Germany. Christian Komusiewicz was supported by a PhD fellowship of the Carl-Zeiss-Stiftung and the DFG, project PABI, NI 369/7.

REFERENCES

[1] E. Alm and A. P. Arkin. Biological networks. *Current Opinion in Structural Biology*, 13(2):193–202, 2003.

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

[3] D. Barrell, E. Dimmer, R. P. Huntley, D. Binns, C. O'Donovan, and R. Apweiler. The GOA database in 2009 - an integrated gene ontology annotation resource. *Nucleic Acids Research*, 37(Database-Issue): 396–403, 2009. (3/19/2010).

[4] 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'08)*, volume 5029 of *LNCS*, pages 31–43. Springer, 2008.

[5] A. Björklund, T. Husfeldt, P. Kaski, and M. Koivisto. Fourier meets Möbius: fast subset convolution. In *Proc. 39th STOC*, pages 67–74. ACM, 2007.

[6] G. Blin, F. Sikora, and S. Vialette. GraMoFoNe: a Cytoscape plugin for querying motifs without topology in Protein-Protein Interactions networks. In *2nd International Conference on Bioinformatics and Computational Biology (BICoB'10)*, pages 38–43, Honolulu, USA, 2010. International Society for Computers and their Applications (ISCA).

[7] E. I. Boyle, S. Weng, J. Gollub, H. Jin, D. Botstein, J. M. Cherry, and G. Sherlock. GO::TermFinder–open source software for accessing gene ontology information and finding significantly enriched gene ontology terms associated with a list of genes. *Bioinformatics*, 20(18):3710–3715, 2004.

[8] S. Bruckner, F. Hüffner, R. M. Karp, R. Shamir, and R. Sharan. Torque: Topology-free querying of protein

interaction networks. *Nucleic Acids Research*, 37(Web Server issue):W106-W108, 2009.

[9] 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.

[10] M. Cesati. Perfect code is W[1]-complete. *Information Processing Letters*, 81:163–168, 2002.

[11] J. Chen and J. Meng. On parameterized intractability: Hardness and completeness. *The Computer Journal*, 51(1):39–59, 2008.

[12] Y. Chen, J. Flum, and M. Grohe. Machine-based methods in parameterized complexity theory. *Theoretical Computer Science*, 339(2-3):167–199, 2005.

[13] 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'07)*, volume 4596 of *WSPC*, pages 27–38. World Scientific, 2007.

[14] R. Dondi, G. Fertin, and S. Vialette. Maximum motif problem in vertex-colored graphs. In *Proc. 20th CPM*, volume 5577 of *LNCS*, pages 221–235. Springer, 2009.

[15] B. Dost, T. Shlomi, N. Gupta, E. Ruppin, V. Bafna, and R. Sharan. Qnet: A tool for querying protein interaction networks. *Journal of Computational Biology*, 15(7):913–925, 2008.

[16] R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer, 1999.

[17] M. R. Fellows. Towards fully multivariate algorithmics: Some new results and directions in parameter ecology. In *Proceedings of the 20th International Workshop on Combinatorial Algorithms (IWOCA'09)*, volume 5874 of *LNCS*, pages 2–10. Springer, 2009.

[18] 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*, 2010. doi: 10.1016/j.jcss.2010.07.003.

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

[20] S. Guillemot and F. Sikora. Finding and counting vertex-colored subtrees. In *Proceedings of the 35th International Symposium on Mathematical Foundations of Computer Science (MFCS'10)*, volume 6281 of *LNCS*, pages 405–416. Springer, 2010.

[21] J. Guo, R. Niedermeier, and S. Wernicke. Parameterized complexity of vertex cover variants. *Theory of Computing Systems*, 41(3):501–520, 2007.

[22] F. Hüffner, S. Wernicke, and T. Zichner. FASPAD: fast signaling pathway detection. *Bioinformatics*, 23 (13):1708–1709, 2007.

[23] F. Hüffner, S. Wernicke, and T. Zichner. Algorithm engineering for color-coding with applications to signaling pathway detection. *Algorithmica*, 52(2):114–132, 2008.

[24] 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.

[25] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, 2006.

[26] R. Niedermeier. Reflections on multivariate algorithmics and problem parameterization. In *Proceedings of the 27th International Symposium on Theoretical Aspects of Computer Science (STACS'10)*, volume 5 of *LIPIcs*, pages 17–32. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2010.

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

[28] SGD project. Saccharomyces genome database. http://downloads.yeastgenome.org/ (3/20/2010).

[29] R. Sharan and T. Ideker. Modeling cellular machinery through biological network comparison. *Nature Biotechnology*, 24:427–433, April 2006.

[30] S. Shen-Orr, R. Milo, S. Mangan, and U. Alon. Network motifs in the transcriptional regulation network of escherichia coli. *Nature Genetics*, 31(1): 64–68, 2002.

[31] S. Tweedie, M. Ashburner, K. Falls, P. Leyland, P. McQuilton, S. Marygold, G. H. Millburn, D. Osumi-Sutherland, A. Schroeder, R. Seal, and H. Zhang. Flybase: enhancing *Drosophila* Gene Ontology annotations. *Nucleic Acids Research*, 37(Database-Issue): 555–559, 2009. (3/22/2010).

[32] S. Wernicke. Efficient detection of network motifs. *IEEE/ACM Transactions of Computational Biology and Bioinformatics*, 3(4):347–359, 2006.

Nadja Betzler studied Bioinformatics at Eberhard-Karls-Universität Tübingen, Germany, where she obtained her diploma in 2006. Since November 2006 she is a Ph.D. student in the group of Prof. Rolf Niedermeier at Friedrich-Schiller-Universität Jena, Germany. Her research interests include parameterized algorithmics, computational complexity, and computational social choice with particular focus on voting systems.

René van Bevern studied Computer Science at Friedrich-Schiller-Universität Jena, Germany, specializing in Theoretical Computer Science and concluding with a diploma degree. Being with the algorithms and complexity group of Prof. Rolf Niedermeier, he is currently working towards his Ph.D. His main research interests include the multivariate complexity analysis of graph and data mining problems.

Michael Fellows received an M.A. in Mathematics and a Ph.D. in Computer Science from the University of California, San Diego. He holds the rank of Professor, and is an Australian Professorial Fellow, with the School of Engineering and Information Technology at Charles Darwin University, in Darwin, Northern Territory, Australia. His main research interests are in algorithms and complexity, where he is known for his foundational work in parameterized complexity and algorithmics. He serves as an Associate Editor with the Journal of Computer and System Sciences, and the ACM Transactions on Algorithms.

Christian Komusiewicz studied Bioinformatics at Friedrich-Schiller-Universität Jena, Germany. Currently, he is working towards his Ph.D. (topic "Algorithmics of Biological Networks") in the group of Prof. Rolf Niedermeier. His research interests are biological network analysis and multivariate algorithmics for NP-hard graph problems. This also includes experimental evaluation of the found algorithms.

Rolf Niedermeier received his diploma degree in Computer Science from TU München and his Ph.D. and habilitation degree from Eberhard-Karls-Universität Tübingen, Germany. He is currently chair for Theoretical Computer Science/Computational Complexity at Friedrich-Schiller-Universität Jena, Germany. His main research interests include algorithms and complexity, with a special focus on multivariate complexity analysis. Application fields of interest include bioinformatics, computational social choice, and graph and network problems in general.