

Efficient Detection of Network Motifs

Sebastian Wernicke

Abstract—Motifs in a given network are small connected subnetworks that occur in significantly higher frequencies than would be expected in random networks. They have recently gathered much attention as a concept to uncover structural design principles of complex networks. Kashtan et al. [*Bioinformatics*, 2004] proposed a sampling algorithm for performing the computationally challenging task of detecting network motifs. However, among other drawbacks, this algorithm suffers from a sampling bias and scales poorly with increasing subgraph size. Based on a detailed analysis of the previous algorithm, we present a new algorithm for network motif detection which overcomes these drawbacks. Furthermore, we present an efficient new approach for estimating the frequency of subgraphs in random networks that, in contrast to previous approaches, does not require the explicit generation of random networks. Experiments on a testbed of biological networks show our new algorithms to be orders of magnitude faster than previous approaches, allowing for the detection of larger motifs in bigger networks than previously possible and thus facilitating deeper insight into the field.

Index terms—Network motif detection algorithm, subgraph enumeration, subgraph sampling, subgraph concentration in random graphs.

1 INTRODUCTION

MANY biological networks¹ appear to contain certain small subnetworks in significantly higher frequencies than random networks. For example, protein-protein interaction networks contain some three- and four-node substructures far more often than one would expect from a random network with similar mathematical properties [19], [36]. Based on the idea that “evolution preserves modules that define specific [...] functions” [31], Milo et al. [23], [24] proposed using such overabundant “topological modules” [31] to uncover the structural design principles of biological networks, thereby coining the term *network motifs* for them.²

Some excitement has surrounded the network motif approach with the original paper by Milo et al. [24] being cited well over 60 times in some major scientific journals as of March 2006. The analysis of network motifs has led to interesting results (of which we name only a few here), e.g., in the areas of protein-protein interaction prediction [1], hierarchical network decomposition [12], and the analysis of temporal gene expression patterns [14], [27], [28]. The transcriptional network of *Escherichia coli* displays motifs to which specific functionalities such as the generation of temporal expression programs or the response to fluctuating external signals can be attributed [24], [28], suggesting that network motifs play key information processing roles in this type of network [15]. The same motifs as in the

transcriptional interaction network of *E. coli* were also identified for the yeast *Saccharomyces cerevisiae*, possibly hinting that common network function implies the sharing of common motifs [18]. Recently, the converse assertion that common motifs imply a common network function has also been made [23].

To put research on network motifs into proper perspective, it should be noted that it has also been met with some criticism. Vázquez et al. [30] demonstrated that global network features such as the clustering coefficient also influence local features such as the abundance of certain subgraphs. Artzy-Randrup et al. [4] found that certain random network models (such as “preferential attachment” as introduced by Barabási and Albert [5]) lead to a display of motifs although there is no explicit selection mechanism for local structures. Milo et al. answer this criticism in [22] by suggesting not only to look at the overabundance of *individual* subgraphs but rather at a broader picture in the form of so-called “subgraph significance profiles.”

Focusing on the algorithmic aspects of network motif detection, this work does not intend to position itself in the discussion about the concept of network motifs. Rather, we present new algorithmic approaches which enable the analysis of larger networks and more complex motifs than previously possible, thus facilitating deeper insight into the field.

1.1 Previous Work

Much work related to network motifs has been spent on interpreting and applying the general concept, but considerably less on the involved algorithmics. Finding network motifs consists of three subtasks:

1. Find which subgraphs occur in the input graph and in which number.
2. Determine which of these subgraphs are topologically equivalent (that is, isomorphic) and group them into subgraph classes accordingly.

1. We use the terms “network” and “node” for fields outside mathematics and computer science. The terms “graph” and “vertex” are used for discussing algorithmic aspects.

2. Note that the term “network motif” has been used in other contexts as well and, e.g., may also refer to a common subnetwork in a set of given networks [26] or to any small labeled subnetwork (without considering connectivity or isomorphy) [8].

• The author is with the Institut für Informatik, Friedrich-Schiller-Universität Jena, Ernst-Abbe-Platz 2, D-07743 Jena, Germany.
E-mail: wernicke@minet.uni-jena.de.

Manuscript received 31 Oct. 2005; revised 24 Mar. 2006; accepted 14 Apr. 2006; published online 31 Oct. 2006.

For information on obtaining reprints of this article, please send e-mail to: tcbb@computer.org, and reference IEEECS Log Number TCBBSI-0127-1005.

3. Determine which subgraph classes are displayed at a much higher frequency than in random graphs under a specified random graph model.

Performing the first subtask by explicitly enumerating all subgraphs of a certain size can be time consuming due to their potentially large number even in small, sparse networks. While some work has been spent on enumerating certain subgraph classes (such as cycles [2]), estimating the frequency of general subgraphs has seemingly attracted less consideration.³ For this reason, Kashtan et al. [15] proposed an algorithm that estimates subgraph occurrences from a randomly sampled set of subgraphs. We discuss this algorithm in full detail in Section 2, mentioning only in passing here that it has a sampling bias which in turn leads to considerable drawbacks such as an inconsistent sampling quality and the need for a computationally expensive bias correction. Besides [15], we are only aware of the work, by Duke et al. [10], on approximating the number of *size-k* subgraphs in a given graph. However, their algorithm—which is based on Szemerédi’s regularity lemma [29]—has no relevance in practice: In order to ensure a reasonable quality of approximation, the input graph has to be astronomically large and contain far more than $e^{k^{65}}$ vertices.

Much work has already been done concerning the second subtask and we rely on McKay’s *nauty* algorithm [20], [21] for performing it in practice.

As to the third subtask, the standard approach for determining subgraph significance so far has been to explicitly generate an ensemble of random graphs (typically at least a thousand) under a given random graph model. One advantage of this approach is clearly that we can rely on a huge selection of existing random graph models which are able to account for, e.g., specific growth mechanisms [3] or built-in topologies [32]. However, independently of the random graph model, the approach of explicit generation is extremely time-consuming since we need to determine subgraph concentrations for each of these graphs just as in the original network. Here, we focus on the popular model of random graphs which preserve the degree sequence of the original network.⁴ While there has been some research concerning the properties of graphs with prescribed degree sequence (such as the average path length [25]), the problem of subgraph distributions within such graphs has only been studied for *size-3* subgraphs in directed sparse random graphs with *expected* degree sequences [13]. Section 3 introduces a new approach to estimate the concentration of any given *size-k* subgraph in random graphs with a given degree sequence without requiring their explicit generation.

1.2 Contribution and Structure of this Work

We provide significant improvements for the first and third subtask of motif detection. Based on a comprehensive analysis of the drawbacks encountered when using the subgraph sampling approach proposed by Kashtan et al. [15], Section 2 presents a new algorithm for subgraph

sampling which does not suffer from these drawbacks. While this comes at the price of only being able to control the *expected* number of samples, our proposed algorithm is faster, much easier to implement, and shows some additional useful features, such as being able to quickly estimate the total number of *size-k* subgraphs in a given graph.

As to the task of determining subgraph significance, Section 3 proposes a new approach that does not require the explicit generation of random graphs with a prescribed degree sequence. This approach leads to a faster algorithm that is, moreover, able to focus on determining the significance of specific subgraphs (which is not possible with previous approaches).

Our algorithms have been implemented in C++; the source code and a user-friendly motif detection tool [33] based on our subgraph sampling algorithm are freely available online at <http://theinf1.informatik.uni-jena.de/motifs/>. For a testbed of biological networks, Section 4 shows that our algorithms detect network motifs by orders of magnitude faster than the implementation of Kashtan et al. This enables the analysis of larger networks and more complex motifs than previously possible.

2 A FASTER ALGORITHM FOR SUBGRAPH SAMPLING

The algorithm for subgraph sampling suggested by Kashtan et al. [15] is based on the idea that we start by selecting a random edge in the input graph and then randomly extend this edge until we obtain a connected subgraph with the desired number of vertices. Section 2.2 discusses this approach and its main drawbacks. We present a new approach to subgraph sampling based on randomized enumeration in Section 2.3.

2.1 Notation

Basic familiarity with graph-theoretic terminology is assumed. For a given graph $G = (V, E)$, we let $n \stackrel{\text{def}}{=} |V|$ and assume that all vertices in V are uniquely labeled by the integers $1, \dots, n$. To abbreviate that the label of a vertex u is larger than that of a vertex v , we write “ $u > v$.” In order to simplify the presentation, edges in $G = (V, E)$ are always identified using set notation, regardless of whether the graph is directed or undirected. In the case that G is directed and contains the edges (u, v) and (v, u) for two vertices u and v , these edges become a single *bidirectional* edge $\{u, v\}$ in our notation.

For a set $V' \subseteq V$ of vertices, its *open neighborhood* $N(V')$ is the set of all vertices from $V \setminus V'$ which are adjacent to at least one vertex in V' . For a vertex $v \in V \setminus V'$, its *exclusive neighborhood* with respect to V' , denoted $N_{\text{excl}}(v, V')$, consists of all vertices neighboring v that do not belong to $V' \cup N(V')$.

A connected subgraph that is induced by a vertex set of cardinality k is called *size-k subgraph*. For a given integer k , the set of all *size-k* subgraphs in G can be partitioned into sets $\mathcal{S}_k^i(G)$ called *subgraph classes*, where two *size-k* subgraphs belong to the same subgraph class if and only if they are isomorphic (that is, if and only if they are topologically

3. Note that the term “frequent subgraphs” is also used in the literature to denote common subgraphs in a set of given graphs (see, e.g., [17]).

4. Other models are also considered in the literature, e.g., additionally preserve the number of bidirectional edges in directed random networks.

Algorithm: EDGE SAMPLING(G, k) (ESA)
Input: A graph $G = (V, E)$ and an integer $2 \leq k \leq |V|$.
Output: Vertices of a randomly chosen size- k subgraph in G .

```

01  $\{u, v\} \leftarrow$  random edge from  $E$ 
02  $V' \leftarrow \{u, v\}$ 
03 while  $|V'| \neq k$  do
04    $\{u, v\} \leftarrow$  random edge between  $V'$  and  $N(V')$ 
05    $V' \leftarrow V' \cup \{u, v\}$ 
06 return  $V'$ 

```

Fig. 1. Pseudocode for the algorithm ESA which samples a random size- k subgraph in a given graph G .

equivalent). The *concentration* $C_k^i(G)$ of a subgraph class $S_k^i(G)$ is defined as

$$C_k^i(G) \stackrel{\text{def}}{=} |S_k^i(G)| \cdot \left(\sum_j |S_k^j(G)| \right)^{-1}.$$

For a graph G , an integer k , and a set \mathcal{R} of size- k subgraphs that were randomly sampled in G by some algorithm \mathcal{A} , a mapping $\hat{C}_k^i : (\mathcal{R}, G) \rightarrow [0, 1]$ is called an *estimator* for $C_k^i(G)$. We say that $\hat{C}_k^i(\mathcal{R}, G)$ is *unbiased* (with respect to \mathcal{A}) if the expected value of $\hat{C}_k^i(\mathcal{R}, G)$ equals $C_k^i(G)$ and *biased* otherwise.⁵

2.2 The Previous Approach: Edge Sampling

For a given graph $G = (V, E)$ and an integer $k \geq 2$, Kashtan et al. [15] suggest sampling a random subgraph by starting with a randomly chosen edge and then adding neighboring vertices until a subgraph of the desired size k is obtained. A pseudocode description of this algorithm, which we will call ESA, is given in Fig. 1.

As already noted in [15], ESA has a bias for sampling certain subgraphs more often than others. Fig. 2 shows a concrete example we have constructed to illustrate this. The total number of connected size-3 subgraphs both in G_1 and G_2 is 28. Since the subgraph \blacktriangle occurs exactly once each in G_1 and G_2 , we should expect that ESA samples \blacktriangle with probability $1/28$ within both graphs. However, we have

$$\Pr[\text{ESA samples } \blacktriangle \text{ in } G_1] = \frac{1}{9} \cdot 1 + \frac{2}{9} \cdot \frac{2}{8} = \frac{1}{6}$$

and

$$\Pr[\text{ESA samples } \blacktriangle \text{ in } G_2] = \frac{3}{12} \cdot \frac{2}{8} = \frac{1}{16}.$$

This illustrates some crucial problems of ESA: The subgraph \blacktriangle is oversampled and—as a direct consequence—the only other occurring size-3 subgraph \blacktriangledown is undersampled. The oversampling of \blacktriangle is worse for G_1 than it is for G_2 and it is possible to show (using an adaptation of the above example) that the amount of bias cannot be estimated simply from the number of edges neighboring the over-sampled subgraph.

5. Sidestepping a formal definition, by “expected value” we mean the average estimated subgraph concentration over a large number of runs of the sampling algorithm \mathcal{A} .

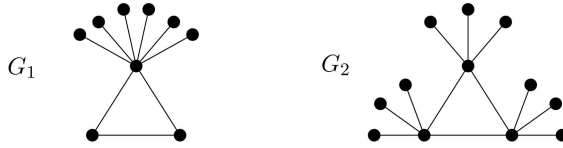


Fig. 2. The graphs G_1 and G_2 have an equal number of (connected) size-3 subgraphs. Furthermore, each of the two graphs has exactly one triangle among its size-3 subgraphs. As outlined in the text, ESA oversamples this subgraph in both G_1 and G_2 , the oversampling being worse for G_1 .

For a given set \mathcal{R} of size- k subgraphs that were randomly sampled using ESA, the sampling bias can be circumvented by using the following unbiased estimator [15]:

$$\hat{C}_k^i(\mathcal{R}, G) \stackrel{\text{def}}{=} \frac{\sum_{G' \in (\mathcal{R} \cap S_k^i(G))} (\Pr[G' \text{ is sampled by ESA}])^{-1}}{\sum_{G' \in \mathcal{R}} (\Pr[G' \text{ is sampled by ESA}])^{-1}}. \quad (1)$$

The main idea here is that each subgraph is (ex post facto) scored inversely proportional to the probability that ESA samples it. While it is possible to correctly estimate $C_k^i(G)$ in this way, several disadvantages remain:

- The bias itself remains as certain subgraphs are still (much) more likely to be sampled than others. This is especially problematic for subgraphs which appear in low concentration and are, at the same time, undersampled by ESA; they are hardly ever found.⁶
- Computing (1) is expensive since the calculation of *each single* probability can require as much as $\mathcal{O}(k^k)$ time [15]. It is also rather complicated to implement as, e.g., care needs to be taken to avoid numerical errors.
- We have no estimate as to what *fraction* of subgraphs has been sampled (which is of interest, e.g., to make statistical estimates about the sampling quality and to determine when enough subgraphs have been sampled).
- ESA can sample the same subgraph multiple times, spending time without gathering new information.

In the next subsection, we suggest a new approach to subgraph sampling that overcomes these problems.

2.3 The New Approach: Randomized Enumeration

The idea here is to start with an algorithm that efficiently enumerates all size- k subgraphs. This algorithm is then modified to randomly skip over some of these subgraphs during its execution, yielding an unbiased subgraph sampling algorithm.

2.3.1 Enumerating All Size- k Subgraphs

Given a graph $G = (V, E)$, the algorithm ESU shown in Fig. 3 enumerates all of its size- k subgraphs. The basic idea of this algorithm is that—starting with a vertex v from the input graph—we add only those vertices to the $V_{\text{Extension}}$ set that have two properties: Their label must be larger than

6. Kashtan et al. [15] observe that ESA can accurately estimate the concentration of $S_k^i(G)$ with less than $(C_k^i(G))^{-1}$ samples for subgraphs which are oversampled. In return, however, other subgraphs might be missed completely for far more than $(C_k^i(G))^{-1}$ samples and would consistently be overlooked as motif candidates.

Algorithm: ENUMERATESUBGRAPHS(G, k) (ESU)

Input: A graph $G = (V, E)$ and an integer $1 \leq k \leq |V|$.

Output: All size- k subgraphs in G .

```

01 for each vertex  $v \in V$  do
02    $V_{\text{Extension}} \leftarrow \{u \in N(\{v\}) : u > v\}$ 
03   call EXTENDSUBGRAPH( $\{v\}, V_{\text{Extension}}, v$ )
04 return

EXTENDSUBGRAPH( $V_{\text{Subgraph}}, V_{\text{Extension}}, v$ )
E1 if  $|V_{\text{Subgraph}}| = k$  then output  $G[V_{\text{Subgraph}}]$  and return
E2 while  $V_{\text{Extension}} \neq \emptyset$  do
E3   Remove an arbitrarily chosen vertex  $w$  from  $V_{\text{Extension}}$ 
E4    $V'_{\text{Extension}} \leftarrow V_{\text{Extension}} \cup \{u \in N_{\text{excl}}(w, V_{\text{Subgraph}}) : u > v\}$ 
E5   call EXTENDSUBGRAPH( $V_{\text{Subgraph}} \cup \{w\}, V'_{\text{Extension}}, v$ )
E6 return

```

Fig. 3. Pseudocode for the algorithm ESU which enumerates all size- k subgraphs in a given graph G . (The definition of the exclusive neighborhood $N_{\text{excl}}(v, V')$ is given in Section 2.1.)

that of v and they may only be neighbored to the newly added vertex w but not to a vertex already in V_{Subgraph} , that is, they must be in the exclusive neighborhood of w with respect to V_{Subgraph} . Some more insight into the structure of ESU can be gained by the following tree structure. (Note that we refer to the vertices of this structure as “nodes” in order to avoid confusion with the vertices of the input graph.)

Definition 1. With a call to EnumerateSubgraphs(G, k), we associate a tree of recursive function calls called ESU-tree. The root located at depth zero represents the function EnumerateSubgraphs(G, k). Each call of ExtendSubgraph($V_{\text{Subgraph}}, V_{\text{Extension}}, v$) is represented by an edge from the node representing the caller function to a node representing the callee. The callee node is labeled $(V_{\text{Subgraph}}, V_{\text{Extension}})$ and located at depth $|V_{\text{Subgraph}}|$.

The structure of the ESU-tree is illustrated in an example in Fig. 4. It is the basis to establish the correctness of ESU in Theorem 2. For a node w in the tree, we use $\text{SUB}(w)$ and $\text{EXT}(w)$ to denote the sets V_{Subgraph} and $V_{\text{Extension}}$ of its label, respectively. Furthermore, it is assumed that the nodes of the ESU-tree are ordered according to the order in which the

subroutines they represent are called. If a node w_1 precedes a node w_2 in this order, we designate this by writing $w_1 \prec w_2$. Given a set of tree nodes, a node is called *minimal* in this set if it precedes all other nodes in the set. The next lemma states some properties of the ESU-tree that are used for proving the correctness of ESU in Theorem 2.

Lemma 1. The ESU-tree has the following properties:

1. Let w_1 be a node distinct from the root. For every vertex $u \in \text{EXT}(w_1)$, the node w_1 has a child node w_2 such that $u \in \text{SUB}(w_2)$.
2. For each node w in the ESU-tree distinct from the root and for each vertex $u \in \text{EXT}(w)$, we have $u > v$, where v is the smallest-label vertex in $\text{SUB}(w)$.
3. Let w_1 and w_2 be two nodes in the tree with a common parent node and $w_1 \prec w_2$. Then, $\text{SUB}(w_1)$ contains exactly one vertex u_1 which is not contained in $\text{SUB}(w_2)$ and vice versa. For every node w' whose path to the root contains w_2 , we have $u_1 \notin \text{SUB}(w')$.

Proof. Property 1 follows from the fact that lines E3 to E5 are carried out for every vertex u in the original $V_{\text{Extension}}$ set (basically, $V_{\text{Extension}}$ can be viewed as a stack from which we pop the vertices u until it is empty).

Property 2 follows directly from lines 02 and E4 of the algorithm, where one condition for a vertex to be added to V_{Subgraph} is that its label is larger than the label of the vertex v . Hence, v , the first vertex added to V_{Subgraph} , is the smallest-label vertex in $\text{SUB}(w)$, as claimed.

For Property 3, two cases need to be considered. In the first case, let the common parent of w_1 and w_2 be the root of the ESU-tree. Then, Property 3 holds because line 03 of ESU is executed exactly once for each vertex v of the input graph and Property 2 ensures that every node w' that is a descendant of w_2 satisfies $\text{SUB}(w') \cap \text{SUB}(w_1) = \emptyset$. Assume now as the second case that the common parent of w_1 and w_2 is distinct from the root. Then, the existence of the vertex u_1 as claimed becomes clear from the fact that once u_1 has been considered for addition to the V_{Subgraph} set, it is removed from $V_{\text{Extension}}$ by line E3 of the ESU algorithm. The claim that once the call to $\text{ExtendSubgraph}(V_{\text{Subgraph}} \cup \{u_1\}, V_{\text{Extension}})$ in line E5 has been completed, no

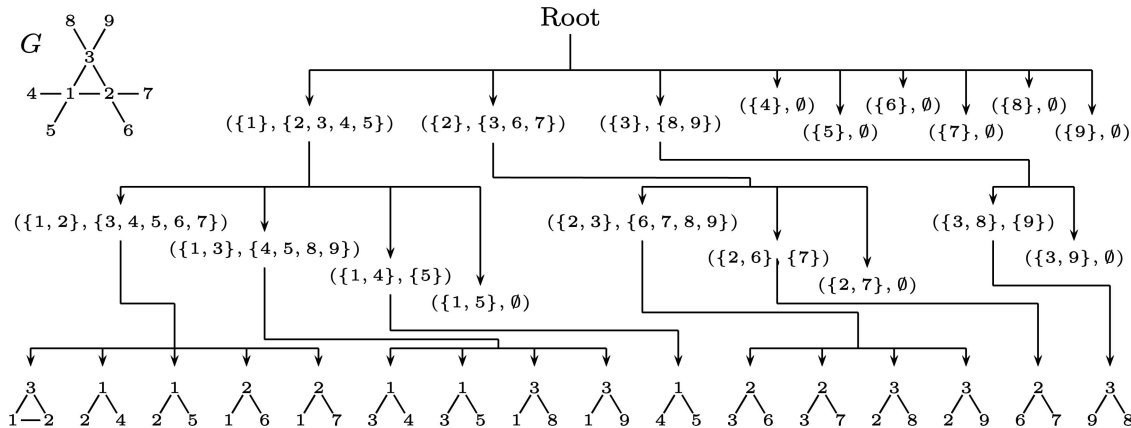


Fig. 4. Given the labeled graph in the left upperhand corner, the above ESU-tree corresponds to calling ENUMERATESUBGRAPHS($G, 3$). The tree has 16 leaves which correspond to the 16 size-3 subgraphs of G .

subgraph containing u_1 is output until we reach line E6 again is proved by observing that u_1 must be neighbor to some vertex in V_{Subgraph} since it is in $V_{\text{Extension}}$. Then, however, there exists no vertex $u' \in V$ for which $u_1 \in N_{\text{ext}}(u', V_{\text{Subgraph}})$ and, hence, once u_1 is removed from $V_{\text{Extension}}$, it is not added to this set by any recursive call of EXTENDSUBGRAPH until we reach line E6 again. \square

Theorem 2. *Given a graph G and $k \geq 2$, ESU enumerates all size- k subgraphs in G . Each size- k subgraph is output exactly once.*

Proof. Given a graph G and an integer $k \geq 2$, we show that every size- k subgraph in G is output at least once and at most once.

“At least once.” Calling EnumerateSubgraphs(G, k), let T be the corresponding ESU-tree. Assume for the purpose of contradiction that there exists a size- k subgraph G' with vertex set $\{v_1, \dots, v_k\}$ in G that is not output by ESU. Without loss of generality, we assume v_1 to be the smallest-label vertex in G' . Because line 03 of the ESU algorithm is called for every vertex $v \in V(G)$ including v_1 , the root of T has exactly one child node w_1 with $\text{SUB}(w_1) = \{v_1\}$. All neighbors of v_1 in G' are in $\text{EXT}(w_1)$ (by line 02 of ESU considering the assumption that v_1 is the smallest-label vertex). Then, by Property 1 of the ESU-tree in Lemma 1, w_1 has a child node w_2 with $\text{SUB}(w_2) = \{v_1, v'\}$ for each neighbor v' of v_1 in G' . Let w'_2 be the minimal of these child nodes and assume without loss of generality that the respective neighbor of v_1 added to $\text{SUB}(w_1)$ is v_2 , that is, $\text{SUB}(w'_2) = \{v_1, v_2\}$. We now claim that $\text{EXT}(w'_2)$ contains all neighbors that v_1 and v_2 have in G' : Assume for the purpose of contradiction that there exists a neighbor which is not contained in $\text{EXT}(w'_2)$. This could only be for three reasons, all of which we can rule out, yielding the desired contradiction:

1. Its label could be smaller than that of v_1 (which can be ruled out because v_1 is the smallest-label vertex in G').
2. It could be neither a neighbor of v_1 nor in the exclusive neighborhood of v_2 (which can be ruled out because, in G' , it is a neighbor of either v_1 , v_2 , or both).
3. It could already have been taken from $\text{EXT}(w'_2)$ (which can be ruled out because we assumed w'_2 to be minimal).

Inductively carrying out the above argument for the vertices v_3, \dots, v_k leads to a leaf node w_k in the ESU-tree for which $\text{SUB}(w_k) = V(G')$, a contradiction to our assumption that G' is not output by the algorithm.

“At most once.” Assume, for the purpose of contradiction, that a subgraph G' is enumerated twice. This means that there are two leaves, w_1 and w_2 , in the corresponding ESU-tree for which $\text{SUB}(w_1) = \text{SUB}(w_2)$. The path p_1 from w_1 to the root must differ at least partly from the path p_2 from w_2 to the root. Call the greatest-depth node in the tree that p_1 and p_2 share the *split node*. Due to Property 3 of Lemma 1, the existence of the split node implies that $\text{SUB}(w_1)$ and $\text{SUB}(w_2)$ differ by at least one element, a contradiction. \square

Besides being useful for the above correctness proof, the ESU-tree exposes some additional useful properties. For example, we can quickly estimate the total number of size- k subgraphs in the input graph using a technique by Knuth [16] that randomly explores paths in the ESU-tree. With this estimate at hand, it is, e.g., possible to see if a total enumeration of subgraphs is expected to be feasible, to estimate the running time of the ESU algorithm (for example, in order to implement a progress indicator [9]) and to make statistical error estimates about the sampling error if no exact enumeration seems feasible. Probably the most important feature of the ESU-tree, however, is that we can use it to efficiently sample subgraphs uniformly at random (that is, without bias). This is further explored in the next subsection.

2.3.2 Uniformly Sampling Size- k Subgraphs.

The ESU algorithm completely traverses its corresponding ESU-tree. Where complete traversal is too time-expensive, we can explore only parts of the ESU-tree such that each leaf is reached with equal probability. For this purpose, a probability $0 < p_d \leq 1$ is introduced for each depth $1 \leq d \leq k$ in the tree. Using p_d , we determine for each child vertex at depth d whether we traverse the subtree rooted at it. This is implemented by replacing lines 03 and E5 of the ESU algorithm with

“With probability p_d , call EXTENDSUBGRAPH(\dots),”

where $d \stackrel{\text{def}}{=} 1$ in line 03 and $d \stackrel{\text{def}}{=} |V_{\text{Subgraph}}| + 1$ in line E5. We call this new algorithm RAND-ESU. To simplify the discussion, we will also use this name when all p_d are set to 1, in which case, RAND-ESU is equivalent to ESU. As we now show, RAND-ESU visits each leaf of the ESU-tree with equal probability and, hence, estimating subgraph concentrations from its output is straightforward.

Lemma 3. *RAND-ESU visits each leaf in the ESU-tree with probability $\prod_d p_d$.*

Proof. Let w_k be a leaf node in the ESU-tree and w_{k-1}, \dots, w_1 the nodes that lie on the path from w_k to the root. Then, we have

$$\begin{aligned} \Pr[w_k \text{ is reached}] &= \Pr[w_k \text{ reached} \mid w_{k-1} \text{ reached}] \\ &\quad \cdot \Pr[w_{k-1} \text{ reached}] \\ &= p_k \cdot \Pr[w_{k-1} \text{ reached}] \\ &= p_k \cdot p_{k-1} \cdot \Pr[w_{k-2} \text{ reached}] \\ &= \dots = p_k \cdot p_{k-1} \cdot \dots \cdot p_1 = \prod_{1 \leq d \leq k} p_d. \end{aligned}$$

\square

Proposition 4. *Given a graph G , an integer k , and $0 < p_d \leq 1$ for $1 \leq d \leq k$, let \mathcal{R} be a set of size- k subgraphs obtained by running RAND-ESU on G using the probabilities p_d . Then,*

$$\hat{C}_k^i(\mathcal{R}, G) \stackrel{\text{def}}{=} \frac{|\{G' \in \mathcal{R} : G' \in \mathcal{S}_k^i(G)\}|}{|\mathcal{R}|}$$

is an unbiased estimator for $C_k^i(G)$.

Proof. The proof follows directly from Lemma 3: If the input graph contains exactly N subgraphs and N' of these are representatives of a subgraph class \mathcal{S}_i^k , the fraction of leaves in the ESU-tree that correspond to representatives of \mathcal{S}_i^k equals N'/N . Since each leaf in the ESU-tree is reached with equal probability, the expected fraction of subgraphs in \mathcal{R} that correspond to representatives of \mathcal{S}_i^k is precisely $N'/N = C_k^i(G)$. \square

It remains to discuss how the values p_d should be chosen. If we wish to sample an expected fraction $0 < q < 1$ of all size- k subgraphs using RAND-ESU, we clearly have to ensure that $\prod_{1 \leq d \leq k} p_d = q$. However, this still leaves us to choose the individual values, that is, do we uniformly set every p_d equal to $q^{1/k}$ or are there better choices? Some general observations are:

- Choosing whether or not to explore a subtree whose root is close to the root of the ESU-tree generally has a higher influence on the total number of explored leaves than for a subtree whose root is farther from it.
- The parameters p_d influence the distribution of the sampling, that is, if p_d is small for small d , some local neighborhoods in the input graph are likely not to be explored at all while others will be explored extensively.
- The running time is influenced from an amortized point of view: The larger the p_d values are for small values of d , the more of the ESU-tree is explored in order to sample a certain expected number of leaves.

Further analysis concerning the choice of sampling parameters p_d can be made based on generating functions [11], [34]: Assume we have two independent random variables X and Y and determine a random variable Z by first choosing a nonnegative integer n according to the distribution of X and then building the sum of n independent random variables chosen according to the distribution of Y . Then, as shown in [11, p. 577], the expected value of Z is

$$\mathbb{E}(Z) = \mathbb{E}(X)\mathbb{E}(Y) \quad (2)$$

with a variance of

$$\text{Var}(Z) = \text{Var}(X)(\mathbb{E}(Y))^2 + \mathbb{E}(X)\text{Var}(Y). \quad (3)$$

Consider a random tree T_k of height k where independently for each node at depth d its number of children is a random variable X_d with expected value $\mathbb{E}_d := \mathbb{E}(X_d)$ and variance $\text{Var}_d := \text{Var}(X_d)$.⁷ For a random traversal of T_k with given parameters p_d , iterative application of (2) and (3) shows that the expected number of visited leaves in T_k is $\mathbb{E}_{T_k} = \prod_{1 \leq d \leq k} (p_d \cdot \mathbb{E}_d)$ (as was to be expected from Lemma 3) with a variance of

7. We are aware that this is only a very coarse model for the ESU-tree, but it allows us to emphasize the main points we wish to make here while it appears that a more precise model does not generate significant additional insight.

$$\text{Var}_{T_k} = \sum_{1 \leq d \leq k} \left(\left(\prod_{1 \leq i < d} p_i \cdot \mathbb{E}_i \right) \cdot (p_d \cdot \text{Var}_d + (p_d - p_d^2) \cdot \mathbb{E}_d) \cdot \left(\prod_{d < i \leq k} (p_i \cdot \mathbb{E}_i)^2 \right) \right). \quad (4)$$

Note how, in accordance with our discussion above, Var_{T_k} is reduced if p_d is small for larger values of d . The term “ $(p_d - p_d^2) \cdot \mathbb{E}_d$ ” found in the middle of (4) is the variance for the number of child nodes that we choose to explore further, its origin lies in lines 03 and E5 of RAND-ESU, which give rise to a binomial distribution for the number of children that are explored. We can reduce this variance as follows: Let w_d be a node in the tree at depth d with x children. Then, instead of deciding independently with probability p_d for each of the x children whether it is to be explored further, we randomly choose x' of the x children, where

$$x' = \begin{cases} \lfloor x \cdot p_d \rfloor & \text{with probability } (x \cdot p_d - \lfloor x \cdot p_d \rfloor) \\ \lfloor x \cdot p_d \rfloor & \text{with probability } (1 - (x \cdot p_d - \lfloor x \cdot p_d \rfloor)) \end{cases}$$

and explore exactly these.⁸ This new procedure has the advantage that it does not change the probability of an individual child being explored ($\Pr[w_{d+1} \text{ is reached} \mid w_d \text{ is reached}] = p_d$ also in this model) while at the same time the variance is reduced from $\mathbb{E}_d \cdot (p_d - p_d^2)$ to at most

$$\max\{(x \cdot p_d - \lfloor x \cdot p_d \rfloor)^2, (\lceil x \cdot p_d \rceil - x \cdot p_d)^2\} < 1.$$

Further analysis will be required to see if any other improvements can be derived for the algorithm. For example, considering Property 2 of Lemma 1, one might consider reducing Var_j in (4) for the ESU algorithm by a certain labeling of the vertices in the input graph.

As a general rule from our analysis in this section, the parameters p_d should be larger for small d and become smaller as d increases—as long as the sacrifice made with respect to the amortized running time per sample is acceptable. This ensures a lower variance for the number of samples and the exploration of many different regions in the input graph.

Concluding this section, while RAND-ESU—as compared to ESA—requires a choice of sampling parameters and only allows for controlling the expected number of samples, it has a lot to offer in return. Most importantly, it is unbiased, which rules out the respective disadvantages of ESA. Also, it is much faster (see the experiments in Section 4) and easier to implement since we do not require any bias-correcting parts. Contrary to ESA, our new algorithm never samples more subgraphs than the input graph contains and its results become exact as the number of samples reaches the total number of size- k subgraphs in the input graph.

8. The idea is to always explore at least $\lfloor x \cdot p_d \rfloor$ children. If $x \cdot p_d$ is not an integer, one more child than $\lfloor x \cdot p_d \rfloor$ is explored with a certain probability, ensuring that exactly $x \cdot p_d$ children are reached on average.

3 DIRECT CALCULATION OF MOTIF SIGNIFICANCE

As already mentioned in the introduction, network motif detection spends considerable amounts of time for the subtask of determining subgraph significance. Traditionally, this task involves explicitly generating an ensemble of random graphs under a certain random graph model and then determining their subgraph concentrations. Here, we propose a new approach to determining subgraph significance that does not need an explicit random graph generation and offers some additional advantages, such as being able to focus the estimation of significance on specific subgraphs.

3.1 A New Approach to Calculating Subgraph Concentrations

We consider the case where the significance of a subgraph is determined by comparing its concentration in the given graph G to its mean concentration $\langle C_k^i(G) \rangle$ in random graphs with the same degree sequence [15], [24]. It is suggested in [15], [24] to estimate $\langle C_k^i(G) \rangle$ by generating a large ensemble of random graphs (typically at least 1,000) with the same degree sequence as the original graph and then sampling subgraphs in these random graphs. We will call this approach EXPLICIT.

The random graphs in EXPLICIT are generated from the original graph by randomly switching endpoints between graph edges. This requires a lot of switching operations while, at the same time, it is never certain when proper randomization has been reached. Also, with this method, we are likely to spend lots of computational effort in estimating the concentrations of subgraph classes we are not interested in.⁹ In this subsection, we propose a new algorithm DIRECT for determining subgraph significance without the need to explicitly generate random graphs. This approach is, moreover, able to focus the estimation of concentrations on specific subgraphs.

Explicitly generating a random network and then estimating its subgraph concentrations can be seen as a random experiment where we first choose a random graph with the same degree sequence as the original graph and then randomly choose subsets of k vertices that induce a connected subgraph. Milo et al. observe that the total number of size- k subgraphs within an ensemble of large graphs with the same degree sequence does not vary much (see supplementary online material to [23] for details). Hence, we could estimate $\langle C_k^i(G) \rangle$ by a differently ordered random experiment where we first select a subset of k vertices and then determine the ratio of graphs with the same degree sequence where these vertices induce a subgraph from $\mathcal{S}_k^i(G)$. More precisely, we have

$$\langle C_k^i(G) \rangle \approx \langle \hat{C}_k^i(G) \rangle \stackrel{\text{def}}{=} \frac{\sum_{G' \in \text{SEQ}(G)} |\mathcal{S}_k^i(G')|}{\sum_{G' \in \text{SEQ}(G)} \sum_i |\mathcal{S}_k^i(G')|}, \quad (5)$$

where $\text{SEQ}(G)$ is the set of all graphs G' that have the same degree sequence as G . Since all graphs G' can be viewed as graphs over the same set of vertices (because they differ

9. This is especially important for sparse networks where a randomly sampled subgraph is likely to be a tree. Trees, however, are often considered to be uninteresting motifs [8].

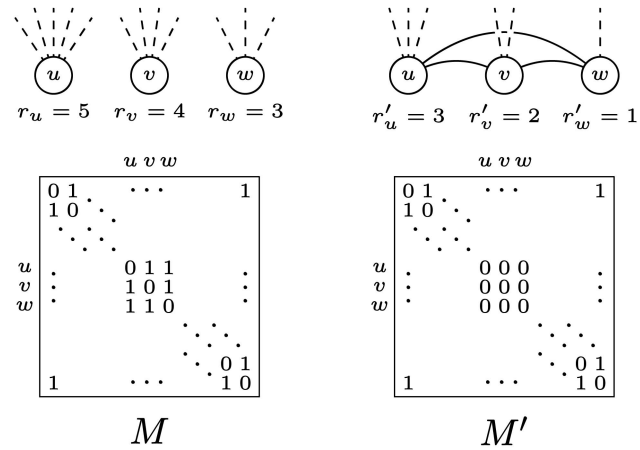


Fig. 5. Theorem 5 allows us to count the number of graphs with a given degree sequence under the constraint that certain edges must be present. On the left, the bitmask matrix M causes every graph with a given degree sequence to be counted by the equation in Theorem 5 since no edge is forbidden by a zero-entry. On the right, we “force” the subgraph to be induced by u , v , and w in all counted graphs by decreasing the degrees of these vertices by two each (e.g., $r_u = 5$ becomes $r'_u = 3$) and using the bitmask matrix M' to avoid the forced edges to be counted.

only in their edge sets), the right part of (5) can also be written as

$$\langle \hat{C}_k^i(G) \rangle = \frac{\sum_{\{v_1, \dots, v_k\} \subseteq V} |\{G' \in \text{SEQ}(G) : G'[\{v_1, \dots, v_k\}] \in \mathcal{S}_k^i\}|}{\sum_{\{v_1, \dots, v_k\} \subseteq V} |\{G' \in \text{SEQ}(G) : G'[\{v_1, \dots, v_k\}] \text{ is connected}\}|}. \quad (6)$$

Both the nominator and denominator of this equation can be estimated in a Monte Carlo approach—that is, by randomly sampling size- k subsets of the vertices in the input graph—as long as we are able to perform the following calculation: Given G and $\{v_1, \dots, v_k\}$, find $|\{G' \in \text{SEQ}(G) : G'[\{v_1, \dots, v_k\}] \in \mathcal{S}_k^i\}|$. As will be discussed in the next subsection, this number can indeed be calculated. We refer to the resulting algorithm as DIRECT throughout the remainder of this work.

3.2 The Number of Graphs that Induce a Fixed Subgraph

In 1974, Bender [6] proved a powerful theorem that allows for an asymptotic estimation of the number of directed graphs with a prescribed degree sequence. This was later extended to a theorem about undirected graphs together with Canfield [7]. The power behind both theorems lies in the fact that we can not only estimate the number of graphs with a prescribed degree sequence, but also, using a *bitmask-matrix* M , specify pairs of vertices that are not to be connected in the graphs we are counting. We describe this in more detail following a formal (and alas quite technical) recapitulation of Bender and Canfield’s theorems.

Definition 2. Given functions $f, g, h : \Omega \rightarrow \mathbb{R}$, we say “ $g \sim h$ uniformly as $f \rightarrow \infty$ ” if

$$\lim_{k \rightarrow \infty} \sup_{\{\omega \in \Omega : f(\omega) = k\}} \left| \frac{g(\omega)}{h(\omega)} - 1 \right| = 0.$$

TABLE 1
Number of Size- k Subgraphs and the Number of Respective Subgraph Classes in Our Test Instances for $3 \leq k \leq 6$

		COLI	YEAST	ELEGANS	YTHAN
	number of nodes	423	688	306	135
	number of edges	519	1 079	2 345	597
	average node degree	1.2	1.6	7.7	4.4
subgraphs	size-3	5 206	13 150	47 322	9 487
	size-4	83 893	183 174	1 394 259	169 733
	size-5	1 433 502	2 508 149	43 256 069	2 908 118
	size-6	22 532 584	32 883 898	1 309 307 357	45 889 039
subgraph classes	size-3	4	7	13	8
	size-4	17	33	197	57
	size-5	83	173	7 071	629
	size-6	390	888	286 375	9 339

All instances are directed graphs.

Theorem 5 (For undirected graphs, adapted from [7]). Let $M = (m_{ij})$ be a binary symmetric $n \times n$ -matrix, where $m_{ii} = 0$ for all i and the number of zeros per row is bounded by a constant. Given a length- n vector $r = (r_1, \dots, r_n)$ over $\{0, 1, \dots, d\}$, let $G(M, r)$ be the number of binary symmetric $n \times n$ -matrices (g_{ij}) that satisfy $m_{ij} = 0 \Rightarrow g_{ij} = 0$ and $\sum_j g_{ij} = r_i$. Then,

$$G(M, r) \sim \frac{\sqrt{2}(f/e)^{(f/2)}}{\exp(a^2 + a + b) \cdot \prod_i (r_i!)}$$

uniformly as $f \rightarrow \infty$, where $a = \sum_i \frac{r_i^2 - r_i}{2f}$, $b = \sum_{m_{ij}=0, i < j} \frac{r_i r_j}{f}$, and $f = \sum_i r_i$. \square

Theorem 6 (For directed graphs, main theorem in [6]). Let $M = (m_{ij})$ be a binary symmetric $n \times n$ -matrix, where $m_{ii} = 0$ for all i and the number of zeros per row is bounded by a constant. Given two length- n vectors $r = (r_1, \dots, r_n)$ and $c = (c_1, \dots, c_n)$ over $\{0, 1, \dots, d\}$, let $G(M, r, c)$ be the number of binary symmetric $n \times n$ -matrices (g_{ij}) that satisfy $m_{ij} = 0 \Rightarrow g_{ij} = 0$, $\sum_j g_{ij} = r_i$, and $\sum_i g_{ij} = c_j$. Then,

$$G(M, r, c) \sim \frac{f!}{\exp(a + b) \cdot \prod_i (r_i!) \cdot \prod_j (c_j!)}$$

uniformly as $f \rightarrow \infty$ where $a = \frac{(\sum_i r_i^2 - r_i) \cdot (\sum_j c_j^2 - c_j)}{2f^2}$, $b = \sum_{m_{ij}=0} \frac{r_i c_j}{f}$, and $f = \sum_i r_i = \sum_j c_j$. \square

Theorems 5 and 6 are both concerned with binary matrices; it is easy to see that they directly relate to the number of adjacency matrices of graphs with a certain given degree sequence. Furthermore, as we now show in more detail, these theorems can also be used to count the total number of graphs with a given degree sequence under the condition that a certain subgraph is fixed. To see this for Theorem 5, observe that we can “forbid” an edge between two vertices v_i and v_j in the counted graphs by setting the corresponding entries m_{ij} and m_{ji} in the bitmask matrix M to zero. We can also “force” an edge between two vertices v_i and v_j by forbidding it via the bitmask matrix M and additionally decreasing the values of r_i and r_j by one each. In this way, Theorem 5 allows us to calculate, for a given degree sequence, how many graphs there are which realize

exactly this degree sequence under the constraint that a certain subgraph is fixed. This is illustrated in Fig. 5 and works similarly for directed graphs using Theorem 6. Given a subgraph class \mathcal{S}_k^i and k vertices $\{v_1, \dots, v_k\}$, we can thus consider all at most $k!$ ways in which these vertices can induce a subgraph from \mathcal{S}_k^i in order to calculate the nominator in (6).

To estimate the denominator in (6), there exist several possibilities. On the one hand, if, for a fixed k , we have calculated the nominator for all subgraph classes \mathcal{S}_k^i , then the denominator is simply the sum of all these values. On the other hand, if we have calculated the nominator of (6) for only a few subgraph classes, then a different approach is possible, the key idea of which is that we do not have to explicitly fix the subgraph between the vertices $\{v_1, \dots, v_k\}$, but only ensure that they are connected. This can be achieved by forcing a spanning tree between the vertices and ensuring—through forbidding certain edges—that no connected subgraph between $\{v_1, \dots, v_k\}$ is considered

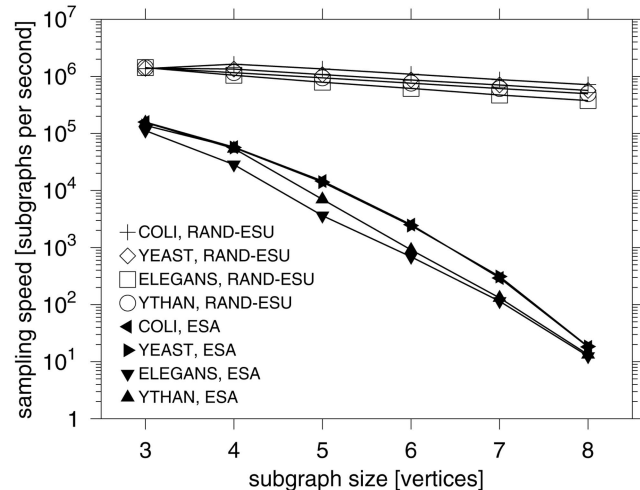


Fig. 6. Sampling speed for different subgraph sizes on a semi-log scale. Independently of the network, ESU (represented by the top four curves) is much faster than ESA and scales much better as the subgraph size increases. Details as to the exact experimental setting are given in the text.

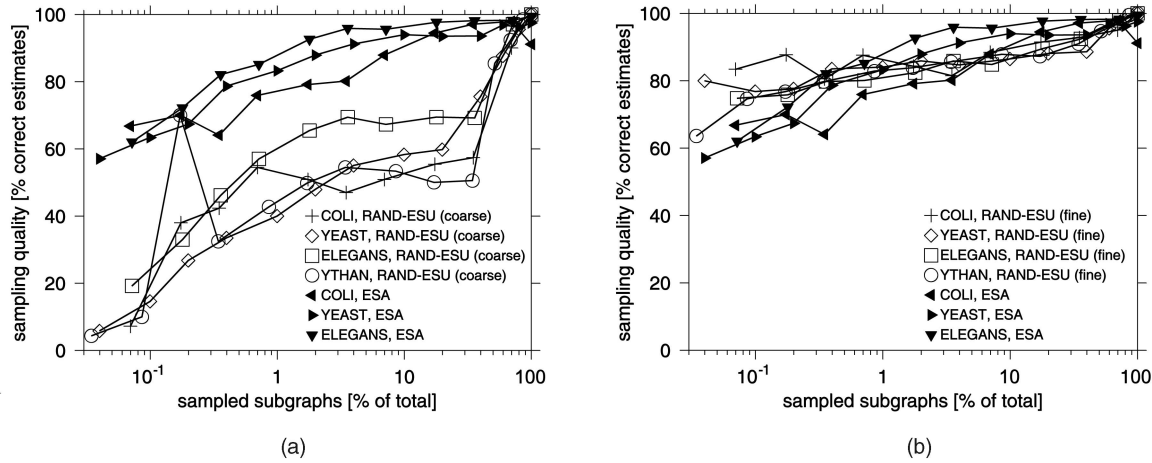


Fig. 7. Sampling quality for size-5 subgraphs (size-4 for YTHAN) versus the percentage of sampled subgraphs (semi-log scale). For our YTHAN instance, the *mfinder* tool reproducibly failed to report results for more than 100 samples, hence this curve is not shown. (a) shows the results for ESA versus RAND-ESU using the sampling parameter setting “coarse” and (b) shows the results using the sampling parameter setting “fine.” Further details as to the experimental setting are given in the text.

twice. To avoid double-counting of certain subgraphs, we make use of the following straightforward observation:

Observation 7. *Every graph with pairwise distinct edge weights has a unique minimum spanning tree.*

Assume that we assign each edge $\{v_i, v_j\}$ (where, without loss of generality, $i < j$) a weight of $i + k \cdot j$. Then, every potential edge in the graph has a unique weight. Hence, we require the enumeration of k^{k-2} trees to estimate the denominator of (6) for undirected graphs, assuming each of these trees to be the minimum spanning tree of the subgraph that is induced by the vertices $\{v_1, \dots, v_k\}$ and forbidding exactly those edges that would contradict this minimality.

For the directed case, the argument is similar to the above, except that, for each spanning tree, we have to consider all ways of coloring the $k-1$ edges with two colors, one color meaning the “directed edge is forced from the vertex with the lower label to the vertex with the higher label” and the other one meaning “the directed edge is forced from the vertex with the higher label to the vertex with the lower label and the directed edge is forbidden from the vertex with the lower label to the vertex with the higher label.” All in all, this requires the consideration of $2^{k-1} \cdot k^{k-2} = 2 \cdot (2k)^{k-2}$ such trees.

At first glance, it might seem as if our new approach to estimating subgraph significance is prohibitively expensive to calculate, but it actually promises a gain in efficiency for a number of reasons:

- If a certain subgraph seldom appears and in relatively few graphs that realize a given degree sequence, a huge number of random graphs have to be generated in order to obtain a sufficient number of samples. In contrast to this, DIRECT always yields a subgraph concentration for any set of vertices where a given subgraph can potentially be induced.
- For small k , we can store all nonautomorph isomorphisms of the respective subgraph (typically

far less than the worst-case $k!$) in an index structure and do not have to recalculate them every time.

- The denominator in (6) is the same for all subgraph classes and, hence, has to be calculated only once. Moreover, it is conceivable that deeper mathematical analysis can estimate this number simply from the degree sequence of the input graph. Even if the denominator is not calculated, we have a *ratio* of subgraph concentrations by the various nominators. Comparing this ratio to the ratio in the original network might already suffice to show the significance of a subgraph as a motif.
- The number of occurring subgraph classes is often far less than the total number of subgraphs (see Table 1) and we can focus our analysis directly on them.

Experiments which are discussed in the next section confirm this expected performance gain.

4 EXPERIMENTAL STUDIES

We have implemented our algorithms from Sections 2 and 3 in C++. The source code is freely obtainable online at <http://theinf1.informatik.uni-jena.de/motifs/>.¹⁰ As a comparison, we used the *mfinder* 1.1 tool¹¹ by Kashtan et al., which implements the ESA algorithm.

4.1 Method and Results














All tests were performed on an AMD Athlon 64 3400+ with 2.4 GHz, 512 KB cache, and 1 GB main memory running under the Debian GNU/Linux 3.1 operating system. Sources were compiled with the GNU gcc/g++ 3.3.4 compiler using the option “-O3.”

The network instances for testing the algorithms were up-to-date versions of the motif detection testbed used by Kashtan et al. [15]. The testbed consists of the instances

10. On the same Web site, we also offer a user-friendly RAND-ESU-based motif detection tool, see [33] for details.

11. Source at <http://www.weizmann.ac.il/mcb/UriAlon/group/NetworkMotifSW.html>.

TABLE 2
Approximate Size-3 Subgraph Concentrations in Random Graphs
Based on the Methods Discussed in Section 3

	COLI		YEAST		ELEGANS		YTHAN	
	$\langle C_k^i(G) \rangle$	$\langle \hat{C}_k^i(G) \rangle$	$\langle C_k^i(G) \rangle$	$\langle \hat{C}_k^i(G) \rangle$	$\langle C_k^i(G) \rangle$	$\langle \hat{C}_k^i(G) \rangle$	$\langle C_k^i(G) \rangle$	$\langle \hat{C}_k^i(G) \rangle$
	9.12 E-1	9.07 E-1	9.00 E-1	8.99 E-1	2.00 E-1	2.00 E-1	4.15 E-1	3.72 E-1
	4.95 E-2	5.11 E-2	7.13 E-2	7.07 E-2	3.65 E-1	3.74 E-1	2.16 E-1	2.28 E-1
	3.65 E-2	4.05 E-2	2.79 E-2	2.93 E-2	3.17 E-1	3.23 E-1	2.17 E-1	2.42 E-1
	2.83 E-4	2.22 E-4	1.43 E-4	1.03 E-4	3.44 E-2	2.74 E-2	4.02 E-2	3.65 E-2
	4.36 E-5	3.20 E-5	1.68 E-5	1.20 E-5	3.69 E-2	2.94 E-2	4.35 E-2	4.66 E-2
	1.44 E-7	7.38 E-8	3.31 E-8	1.72 E-8	2.49 E-3	1.58 E-3	4.05 E-3	3.67 E-3
	1.44 E-3	1.33 E-3	1.09 E-3	1.06 E-3	3.23 E-2	3.45 E-2	4.79 E-2	5.29 E-2
	2.90 E-6	3.33 E-6	9.46 E-7	9.94 E-7	4.08 E-3	4.45 E-3	1.71 E-3	2.59 E-3
	2.11 E-6	1.60 E-6	1.34 E-6	9.52 E-7	2.13 E-3	1.79 E-3	3.47 E-3	3.36 E-3
	7.60 E-7	4.59 E-7	1.95 E-7	1.34 E-7	1.65 E-3	1.43 E-3	6.04 E-3	6.73 E-3
	1.98 E-7	1.50 E-7	5.38 E-8	3.71 E-8	2.41 E-3	2.07 E-3	3.21 E-3	3.94 E-3
	3.82 E-9	1.72 E-9	6.80 E-10	3.26 E-10	5.78 E-4	4.12 E-4	1.59 E-3	1.58 E-3
	—	1.73 E-12	—	1.96 E-13	2.87 E-5	1.68 E-5	9.76 E-5	7.74 E-5

The concentrations $\langle C_k^i(G) \rangle$ were determined with EXPLICIT by generating 10 million random graphs with the same degree sequence as the original network (observe that, even with this large number of random graphs, the subgraph shown in the bottom-most row of the table was never found in COLI and YEAST although the degree sequences of both networks theoretically allow for its presence). The concentrations $\langle \hat{C}_k^i(G) \rangle$ were calculated by DIRECT, evaluating (6) from Section 3.2 for 100 million random size-3 vertex subsets.

COLI (transcriptional network of *Escherichia coli* [28]), YEAST (transcriptional network of *Saccharomyces cerevisiae* [24]), ELEGANS (neuronal network of *Caenorhabditis elegans* [15]), and YTHAN (food web of the Ythan estuary [35]). Some properties of these networks are summarized in Table 1.

In order to compare RAND-ESU with ESA for speed, we measured the sampling speed of both algorithms on the testbed instances for $3 \leq k \leq 8$. Since the relative sampling speed of RAND-ESU depends on the sampling parameters (recall the discussion in Section 2.3.2), we determined the speed of RAND-ESU to be its mean speed for three different settings of (p_1, \dots, p_k) that lead to sampling of an expected 10 percent of all subgraphs, namely, $(1, \dots, 1, .316, .316)$, $(1, \dots, 1, .5, .2)$, and $(1, \dots, 1, .1)$. The results are shown in Fig. 6. The speed of the deterministic ESU algorithm is not shown in the figure; it proved to be slightly faster than that of RAND-ESU. Note that, in order to get comparable results, our time measurements do not include the grouping of sampled subgraphs into nonisomorphic classes because *mfinder* uses a much less efficient canonical labeling routine than the *nauty* algorithm employed in our implementation.

To compare RAND-ESU with ESA for sampling quality, we formally define the sampling quality to be the percentage of subgraph classes \mathcal{S}_k^i for which C_k^i is estimated with at most 20 percent relative error. In doing so, for a given number of subgraph samples, we consider only those subgraph classes for the quality measurement that we would expect to sample at least 10 times. The reason for this is that, in a real-case scenario, one would not want to rely on too few sampled subgraphs to estimate the overall concentration of the respective class. As in the speed measurements, RAND-ESU

was run with different settings of the sampling parameters (p_1, \dots, p_k) in order to sample an expected percentage p of subgraphs in the input network. We refer to these settings as “coarse” $(1, \dots, 1, \sqrt{p}, \sqrt{p})$ and “fine” $(1, \dots, 1, p)$.¹² The obtained results are shown in Fig. 7.

As to the direct calculation of subgraph significance introduced in Section 3, two sets of experiments were carried out for the four testbed instances. The first set measures how close the subgraph concentrations determined by EXPLICIT are to those calculated by DIRECT. For this, we calculated the subgraph concentration of all 13 nonisomorphic directed size-3 subgraphs both by explicitly generating 10 million random graphs as well as by sampling 100 million size-3 subsets of vertices. (We used the rather small value of $k = 3$ in order to make such a large number of samples feasible and because this enables us to gain an overview over *all* subgraph classes.) The results are given in Table 2.

To compare the speed of DIRECT with EXPLICIT, we measured the standard deviation of the output subgraph concentrations with respect to time.¹³ Since both algorithms converge to slightly different values—as shown in Table 2—the standard deviations have been made comparable by normalizing them to $\langle C_k^i(G) \rangle$ and $\langle \hat{C}_k^i(G) \rangle$ for DIRECT and EXPLICIT, respectively. A representative subset of the obtained results is given in Fig. 8; they are further discussed in the next section.

¹² Note how we also used these settings in the speed measurements.

¹³ Measuring the standard deviation with respect to some machine-independent variable such as “number of samples” does not seem feasible due to the very different nature of the two algorithms.

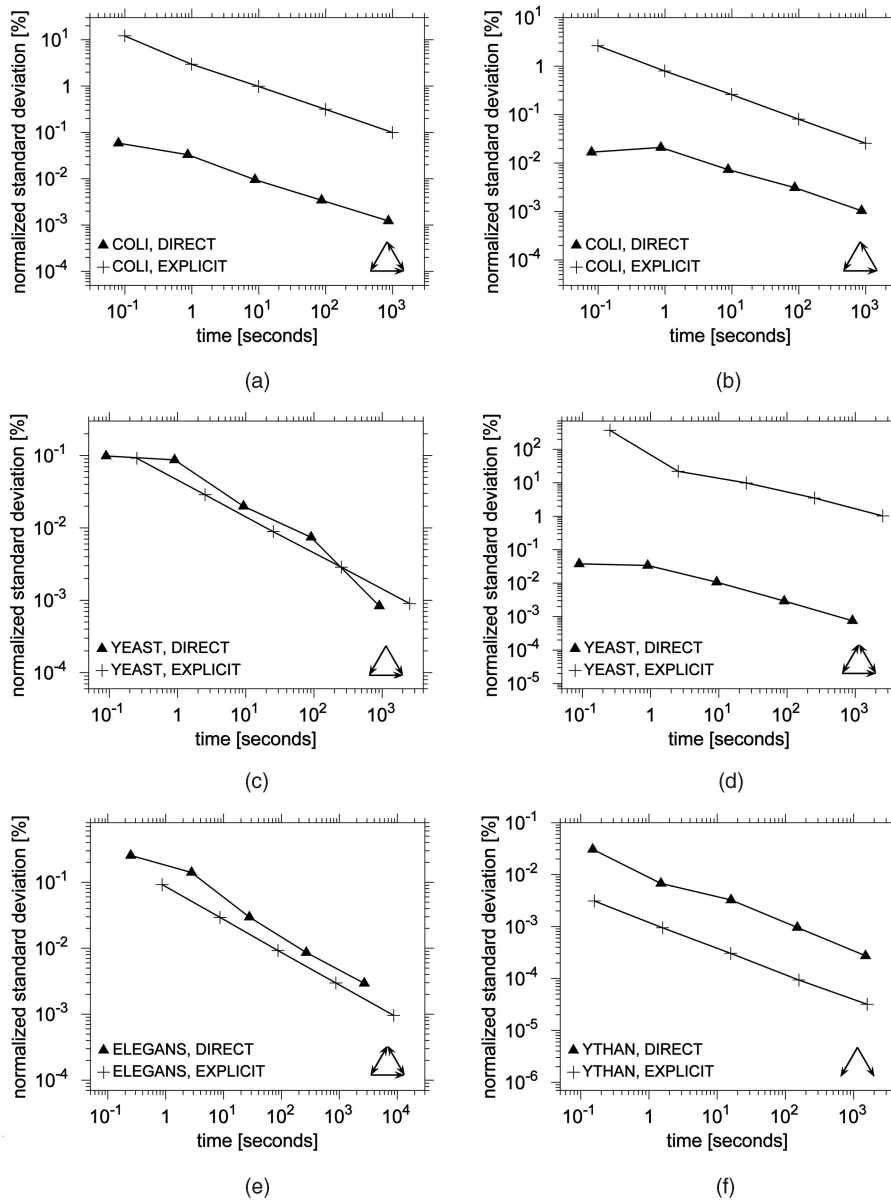


Fig. 8. Representative examples for the speed comparison of DIRECT with EXPLICIT. The name of the network is given in the lower lefthand corner, the subgraph class in the lower righthand corner of each graph. Details as to the experimental setting are given in the text.

4.2 Discussion

Most notable in Fig. 6, RAND-ESU is much faster than the ESA sampling in *mfinder*. This amounts to several orders of magnitude for larger subgraphs ($k \geq 5$). For small sampling quantities, the “coarse” variant of RAND-ESU proved to be faster than the “fine” variant (not explicitly shown in Fig. 6). However, Fig. 7a shows that the resulting sampling quality from using “coarse” settings for the p_d values is relatively low when compared to that of ESA. The qualities are roughly equal for the “fine” variant (see Fig. 7b) with ESA having a slight advantage for sampling sizes above 1 percent and close to 100 percent. (Note that, for 100 percent, RAND-ESU is equivalent to ESU and the results are exact.) Two things are to be noted in this respect, though: First, RAND-ESU is much faster and can, e.g., fully enumerate all size-5 subgraphs in roughly the same time that ESA needs to sample 1 percent of them. Second, the sampling quality of

the “fine” variant appears to be more consistent for different networks, e.g., in some percentage ranges, ESA has a very good sampling quality for ELEGANS and a comparably fair one for COLI, whereas the “fine” RAND-ESU remains much more consistent here. Also note that—contrary to ESA—statistical estimates about the achieved sampling quality can be made with RAND-ESU because of its unbiasedness (especially with the “fine” variant where individual samples are fully independent of each other) and the ability to estimate the total number of subgraphs. Contrary to ESA, the sampling quality of RAND-ESU becomes perfect as the percentage of sampled subgraphs reaches 100 percent.

As to the estimation of subgraph significance, Table 2 shows that DIRECT yields subgraph concentrations that are mostly very close to those output by EXPLICIT. These concentrations generally appear to be closer for the denser

instances ELEGANS and YTHAN than for the sparse instances COLI and YEAST with most of the significant deviations occurring for subgraph classes that appear in very low concentrations of $\langle C_k^i(G) \rangle < 10^{-6}$ (observe from Table 1 that this is far less than one over the total number of size-3 subgraphs in these networks). Overall, the direct calculation of subgraph significance thus does not seem to have an impact on which subgraphs in the given network would be classified as motifs, making DIRECT a valid tool for determining subgraph significance.

As to the convergence speed of EXPLICIT and DIRECT, our experiments show, on the one hand, that EXPLICIT is generally faster than DIRECT for subgraphs which occur in high concentrations. Examples for this are given in Fig. 8c, Fig. 8e, and Fig. 8f. This particular speed advantage of EXPLICIT does not appear to be relevant for practical purposes, however, because both algorithms quickly reach a very low normalized standard deviation ($< 1\%$). On the other hand, as is exemplarily illustrated in Fig. 8a, Fig. 8b, and Fig. 8d, DIRECT is considerably more efficient than EXPLICIT for subgraphs which have a very low concentration in random graphs. Here, an acceptable standard deviation is achieved many orders of magnitude faster. With our new approach, it is even possible to estimate the frequency of some subgraphs for which EXPLICIT does not give any results due to an extremely low average concentration in the explicitly generated random graphs (two examples for this are given in the bottommost row of Table 2). Summarizing, it seems justified to say that DIRECT provides an accurate and much faster alternative to EXPLICIT for determining subgraph significance if the random graph model is that of preserved degree sequence.

5 CONCLUSION

Based on a detailed analysis of previous approaches, we have presented new algorithmic techniques which allow for a faster detection of network motifs and offer useful additional features such as unbiased subgraph sampling and a specifically targeted detection of subgraph significance. This enables motif detection for larger networks and more complex motifs than was previously possible, hopefully facilitating future research on network motifs and its adjoining fields in systems biology.

Further research could improve the presented sampling technique, e.g., by examining how the labeling of the vertices in the input graph affects the sampling quality or seeing if RAND-ESU can be tweaked to selectively sample "interesting" parts of the input graph. For subgraph significance, we have shown that a direct calculation scheme may serve as a fast and accurate alternative to the explicit generation of random networks. It would be interesting to further explore this path by extending the scheme to classes of random background models other than those that solely preserve the degree sequence.

ACKNOWLEDGMENTS

This research was supported by the Deutsche Telekom Stiftung and the Studienstiftung des deutschen Volkes. The author is grateful to Rolf Niedermeier (Jena), Jens Gramm

(Tübingen), and Falk Hüffner (Jena) for helpful discussions and comments. An anonymous referee of WABI 2005 also made some insightful and useful remarks on an extended abstract of this work which appears under the title "A Faster Algorithm for Detecting Network Motifs" in the *Proceedings of the Fifth Workshop on Algorithms in Bioinformatics* (WABI '05), pp. 165-177, October 2005. Funded by the Deutsche Forschungsgemeinschaft project PEAL (Parameterized Complexity and Exact Algorithms), NI 369/1, Florian Rasche (Jena) greatly helped in extending the functionality and usability of the ESA implementation in order to make it a user-friendly motif detection tool.

REFERENCES

- [1] I. Albert and R. Albert, "Conserved Network Motifs Allow Protein-Protein Interaction Prediction," *Bioinformatics*, vol. 20, no. 18, pp. 3346-3352, 2004.
- [2] N. Alon, R. Yuster, and U. Zwick, "Finding and Counting Given Length Cycles," *Algorithmica*, vol. 17, no. 3, pp. 209-223, 1997.
- [3] L.A.N. Amaral, A. Scala, M. Barthélemy, and H.E. Stanley, "Classes of Small-World Networks," *Proc. Nat'l Academy of Sciences*, vol. 97, no. 21, pp. 11149-11152, 2000.
- [4] Y. Artzy-Randrup, S.J. Fleishman, N. Ben-Tal, and L. Stone, "Comment on 'Network Motifs: Simple Building Blocks of Complex Networks' and 'Superfamilies of Designed and Evolved Networks'," *Science*, vol. 305, no. 5687, p. 1007c, 2004.
- [5] A.L. Barabási and R. Albert, "Emergence of Scaling in Random Networks," *Science*, vol. 286, no. 5439, pp. 509-512, 1999.
- [6] E.A. Bender, "The Asymptotic Number of Non-Negative Matrices with Given Row and Column Sums," *Discrete Math.*, vol. 10, no. 2, pp. 217-223, 1974.
- [7] E.A. Bender and E.R. Canfield, "The Asymptotic Number of Labeled Graphs with Given Degree Sequences," *J. Combinatorial Theory Series A*, vol. 24, no. 3, pp. 296-307, 1978.
- [8] J. Berg and M. Lässig, "Local Graph Alignment and Motif Search in Biological Networks," *Proc. Nat'l Academy of Sciences*, vol. 101, no. 41, pp. 14689-14694, 2004.
- [9] D.A. Berque, M.K. Goldberg, and J.A. Edmonds, "Implementing Progress Indicators for Recursive Algorithms," *Proc. Fifth ACM-SIGAPP Symp. Applied Computing (SAC '93)*, pp. 533-538, 1993.
- [10] R.A. Duke, H. Lefmann, and V. Rödl, "A Fast Approximation Algorithm for Computing the Frequencies of Subgraphs in a Given Graph," *SIAM J. Computing*, vol. 24, no. 3, pp. 598-620, 1995.
- [11] R.L. Graham, D.E. Knuth, and O. Patashnik, *Concrete Mathematics: A Foundation for Computer Science*, second ed. Addison-Wesley, 1994.
- [12] S. Itzkovitz et al., "Coarse-Graining and Self-Dissimilarity of Complex Networks," *Physical Rev. E*, vol. 71, p. 016127, 2005.
- [13] S. Itzkovitz et al., "Subgraphs in Random Networks," *Physical Rev. E*, vol. 68, p. 026127, 2003.
- [14] S. Kalir et al., "Ordering Genes in a Flagella Pathway by Analysis of Expression Kinetics from Living Bacteria," *Science*, vol. 292, no. 5524, pp. 2080-2083, 2001.
- [15] N. Kashtan, S. Itzkovitz, R. Milo, and U. Alon, "Efficient Sampling Algorithm for Estimating Subgraph Concentrations and Detecting Network Motifs," *Bioinformatics*, vol. 20, no. 11, pp. 1746-1758, 2004.
- [16] D.E. Knuth, "Estimating the Efficiency of Backtrack Programs," *Selected Papers on Analysis of Algorithms*, Stanford Junior Univ., Palo Alto, Calif., 2000.
- [17] M. Koyutürk, A. Grama, and W. Szpankowski, "An Efficient Algorithm for Detecting Frequent Subgraphs in Biological Networks," *Bioinformatics*, vol. 20, no. supplement 1, pp. i200-i207, 2004.
- [18] T.I. Lee et al., "Transcriptional Regulatory Networks in *Saccharomyces Cerevisiae*," *Science*, vol. 298, no. 5594, pp. 799-804, 2002.
- [19] S. Li et al., "A Map of the Interactome Network of the Metazoan *C. Elegans*," *Science*, vol. 303, no. 5657, pp. 540-543, 2004.
- [20] B.D. McKay, "Practical Graph Isomorphism," *Congressus Numerantium*, vol. 30, pp. 45-87, 1981.

- [21] B.D. McKay, "Nauty User's Guide (Version 1.5)," Technical Report TR-CS-90-02, Dept. of Computer Science, Australian Nat'l Univ., 1990.
- [22] R. Milo et al., "Response to Comment on 'Network Motifs: Simple Building Blocks of Complex Networks' and 'Superfamilies of Designed and Evolved Networks'," *Science*, vol. 305, no. 5687, p. 1007d, 2004.
- [23] R. Milo et al., "Superfamilies of Designed and Evolved Networks," *Science*, vol. 303, no. 5663, pp. 1538-1542, 2004.
- [24] R. Milo et al., "Network Motifs: Simple Building Blocks of Complex Networks," *Science*, vol. 298, no. 5594, pp. 824-827, 2002.
- [25] M.E.J. Newman, S.H. Strogatz, and D.J. Watts, "Random Graphs with Arbitrary Degree Distributions and Their Applications," *Physical Rev. E*, vol. 64, p. 026118, 2001.
- [26] S. Ott, A. Hansen, S. Kim, and S. Miyano, "Superiority of Network Motifs over Optimal Networks and an Application to the Revelation of Gene Network Evolution," *Bioinformatics*, vol. 21, no. 2, pp. 227-238, 2005.
- [27] M. Ronen, R. Rosenberg, B.I. Shraiman, and U. Alon, "Assigning Numbers to the Arrows: Parameterizing a Gene Regulation Network by Using Accurate Expression Kinetics," *Proc. Nat'l Academy of Sciences*, vol. 99, no. 16, pp. 10555-10560, 2002.
- [28] S.S. Shen-Orr, R. Milo, S. Mangan, and U. Alon, "Network Motifs in the Transcriptional Regulation Network of Escherichia Coli," *Nature Genetics*, vol. 31, no. 1, pp. 64-68, 2002.
- [29] E. Szemerédi, "Regular Partitions of Graphs," *Problèmes Combinatoires et Théorie des Graphes*, no. 260 in Colloques internationaux CNRS, pp. 399-401, 1976.
- [30] A. Vázquez et al., "The Topological Relationship between the Large-Scale Attributes and Local Interaction Patterns of Complex Networks," *Proc. Nat'l Academy of Sciences*, vol. 101, no. 52, pp. 17940-17945, 2004.
- [31] A. Vespignani, "Evolution Thinks Modular," *Nature Genetics*, vol. 35, no. 2, pp. 118-119, 2003.
- [32] D.J. Watts, *Small Worlds: The Dynamics of Networks between Order and Randomness*. Princeton Univ. Press, 1999.
- [33] S. Wernicke and F. Rasche, "FANMOD: A Tool for Fast Network Motif Detection," *Bioinformatics*, vol. 22, no. 9, pp. 1152-1153, 2006.
- [34] H.S. Wilf, *Generatingfunctionology*, second ed. Academic Press, 1994.
- [35] R.J. Williams and N.D. Martinez, "Simple Rules Yield Complex Food Webs," *Nature*, vol. 404, no. 6774, pp. 180-183, 2000.
- [36] S.H. Yook, Z.N. Oltvai, and A.L. Barabási, "Functional and Topological Characterization of Protein Interaction Networks," *Proteomics*, vol. 4, no. 4, pp. 928-942, 2004.



Sebastian Wernicke studied bioinformatics at the Eberhard-Karls-Universität in Tübingen, Germany, from which he graduated in 2003. Currently, he is a PhD student of Rolf Niedermeier, chair for theoretical computer science and computational complexity, at the Friedrich-Schiller-Universität in Jena, Germany. His research focuses on parameterized and exact algorithms for hard combinatorial problems in computational biology.

► For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/publications/dlib.