

Approximating the Number of Network Motifs

Mira Gonen and Yuval Shavitt

Abstract. The World Wide Web, the Internet, coupled biological and chemical systems, neural networks, and social interacting species are only a few examples of systems comprising a large number of highly interconnected dynamical units. These networks contain characteristic patterns, *network motifs*, that occur far more often than in randomized networks with the same degree sequence. Several algorithms have been suggested for counting or detecting the number of occurrences of network motifs as trees and bounded treewidth subgraphs of size $O(\log n)$, at most 7 for some motifs. In addition, *local motif counting*, counting the number of motifs in which a node participates, was recently suggested as a method of classifying nodes in the network. The premise is that the distribution of motifs in which a node participates is an indication of its function in the network. Therefore, local counting of network motifs provides a major challenge. However, no such practical algorithm exists other than local counting of triangles. We present several algorithms with time complexity $O(((3e)^k \cdot n \cdot |E| \cdot \log \frac{1}{\delta})/\epsilon^2)$ that approximate for every vertex the number of occurrences of the motif in which the vertex participates, for k -length cycles and k -length cycles with a chord, where $k = O(\log n)$, and algorithms with time complexity $O((n \cdot |E| \cdot \log \frac{1}{\delta})/\epsilon^2 + |E|^2 \cdot \log n + |E| \cdot n \log n)$ that approximate for every vertex the number of noninduced occurrences of the motif in which the vertex participates for all motifs of size four. In addition, we show algorithms that approximate the total number of occurrences of these network motifs when no efficient algorithm exists. Some of our algorithms use the “color-coding” technique.

I. Introduction

I.1. Background and Motivation

The World Wide Web, the Internet, coupled biological and chemical systems, neural networks, and social interacting species are only a few examples of sys-

tems comprising a large number of highly interconnected dynamical units. A first approach to capturing the global properties of such systems is to model them as graphs whose nodes represent the dynamical units and whose links stand for the interactions between them. Such networks have been extensively studied by exploring their global topological features such as power-law degree distribution, the existence of a dense core, and small diameter [Faloutsos et al. 99, Albert and Barabási 00, Chen et al. 02, Lakhina et al. 03, Willinger et al. 02, Winick and Jamin 02, Bu and Towsley 02, Bar et al. 05, Bar et al. 04, Feldman and Shavitt 08, Shavitt and Shir 05, Li and Chen 03, Bianconi and Barabási 01, Barford et al. 01, Reittu and Norros 04, Tauro et al. 01, Subramanian et al. 02, Tangmunarunkit et al. 02, Govindan and Tangmunarunki 07, Brunet and Sokolov 02, Sagie and Wool 04, Siganos et al. 06]. However, two networks that have similar global features, such as similar degree sequences, can have significant differences in structure, which can be captured by examining the local structures they include: for instance, one of them may include a specific subgraph many more times than the other. Therefore, it has been suggested that such small subgraphs, termed *network motifs*, may be elementary building blocks that carry out key functions in the network. The authors of [Milo et al. 02] have found motifs in networks from biochemistry, neurobiology, ecology, and the World Wide Web. Moreover, [Hales and Arteconi 08] presents results from a motif analysis of networks produced by peer-to-peer protocols. The authors show that the motif profiles of such networks closely match protein-structure networks. Thus efficiently detecting and counting the number of network motifs is a major challenge. As a result, novel computational tools have been developed for counting subgraphs in a network and discovering network motifs.

There are quite a few works that deal with finding subgraphs of a certain kind and counting their number. One of the most elegant techniques devised is *color coding*, introduced in [Alon et al. 95] and further applied in [Alon et al. 97, Arvind and Raman 02, Alon and Gutner 07, Alon et al. 08, Dost et al. 07, Shlomi et al. 06]. Color coding is an innovative combinatorial approach that was introduced to detect simple paths, trees, and bounded treewidth subgraphs in unlabeled graphs. Color coding is based on assigning random colors to the vertices of an input graph. Then, only the subgraphs each of whose vertices has a unique color (termed “colorful” subgraphs) are efficiently counted using dynamic programming, in time polynomial with n , the size of the input graph. Alon et al. showed that by repeating this procedure sufficiently many times (polynomial with n , provided that the subgraph we are looking for is of size $O(\log n)$) it is guaranteed that a specific occurrence of the query subgraph will be detected with high probability. The color-coding technique is a building block in some of the algorithms presented in this paper. [Arvind and Raman 02] uses

color coding and a technique from [Karp and Luby 83] to design a randomized algorithm for approximately counting the number of subgraphs in a given graph G that are isomorphic to a bounded treewidth graph H . The running time of the algorithm is $k^{O(k)} \cdot n^{b+O(1)}$, where n and k are the numbers of vertices in G and H , respectively, and b is the treewidth of H . [Alon and Gutner 07] uses color coding and balanced families of perfect hash functions to obtain a deterministic algorithm for counting simple paths or cycles of size k in time $2^{O(k \log \log k)} n^{O(1)}$. These results are improved in [Alon et al. 08] in terms of the dependence on k .

[Przulj et al. 05] offers a description of how to count all induced subgraphs with up to five vertices in a PPI (protein–protein interaction) network.¹ [Hormozdiari et al. 07] develops faster techniques for counting induced subgraphs of size up to 6, and [Grochow and Kellis 07] develops such techniques for size up to 7. The running times of these techniques all increase exponentially with the size of the motif. [Kashtan et al. 04] shows an algorithm for detecting induced network motifs that sample the network. This algorithm detects induced occurrences of small motifs (motifs with $k \leq 7$ vertices). [Wernicke 06] claims that Kashtan et al.’s algorithm suffers from a sampling bias and scales poorly with increasing subgraph size. Thus, [Wernicke 06] presents an improved algorithm for network motif detection that overcomes these drawbacks. [Scott et al. 05] focuses on the subgraph detection problem. [Dost et al. 07] shows how to solve the subgraph detection problem for subgraphs of size $O(\log n)$, provided that the query subgraph is a simple path, a tree, or a bounded treewidth subgraph. [Duke et al. 95] gives an algorithm that provides the number of induced copies of certain subgraphs, with a bounded error. This algorithm has running time of order $O(n^{1/\log \log n} \cdot M(n))$, where $M(n)$ is the time needed to square an $n \times n$ matrix with $(0, 1)$ -entries over the integers. [Itzhack et al. 07] presents an algorithm based on network decomposition via node removal for counting k -size network motifs in large networks, where k is 3 or 4. This algorithm detects all motifs containing a given node by measuring all its incoming and outgoing neighbors of degree $k - 1$, and then removing this node. The algorithm has a constant memory cost, a CPU cost that is linear with the number of counted motifs, and is faster than previous full-enumeration algorithms.

[Bianconi and Capocci 03] gives an analytic expression for the number of cycles of a certain size as a function of the system size in the Barabási–Albert network. [Bianconi and Marsili 05] evaluates the average number of cycles in random scale-free networks. The authors show that the most frequent size of a cycle in a scale-free network is of order the network’s size. Moreover, they indicate that

¹Note that G_0 is an induced subgraph of a graph G if and only if for each pair of vertices v_0 and w_0 in G_0 and their corresponding vertices v and w in G there is an edge between v_0 and w_0 in G_0 if and only if there is an edge between v and w in G .

small-length cycles are more frequent when the second moment of the degree distribution diverges. [Marinari et al. 07] analyzes the problem of discovering long cycles in random graphs. The authors propose and test two algorithms for this task. The first is based on a message-passing procedure, and the second follows a Monte Carlo Markov chain strategy. [Bianconi and Marsili 06] counts the average number of cliques in random scale-free networks when the network is large. The authors show that in contrast to Erdős–Renyi graphs, cliques appear also when the average degree is finite. In addition, the authors prove that in random scale-free networks the clique number diverges with the system size.

[Gonen et al. 10] introduces a sublinear algorithm for approximating the number of constant-size stars. The algorithm is based on querying parts of the graph. The authors show that their result is tight up to polylogarithmic factors in n and the dependence in ϵ (the counting error) by giving a matching lower bound. They also prove negative results for sublinear counting of length-3 paths and triangles.

A new systematic measure of a network’s *local* topology was recently suggested in [Przulj 07]. The authors term this measure “graphlet distribution” of a vertex. Namely, they count for each vertex the number of all motifs of size at most five in which the vertex participates. [Gordon et al. 10] discusses local motif counting as a method of classifying nodes in the network. The premise is that the distribution of motifs in which a node participates is an indication of its function in the network; thus nodes can be divided into functional classes. [Van Kerrebroeck and Marinari 08] suggests using the number of cycles in which a vertex participates as a method to quantify the role of vertices in the network. [Becchetti et al. 08] shows that the distribution of the local number of triangles and the related clustering coefficient can be used to detect the presence of spamming activity in large-scale web graphs, as well as to provide useful features for the analysis of biochemical networks or the assessment of content quality in social networks. However, no practical algorithm (namely with a running time below $O(n^k)$, where k is the graphlet size) for local counting of other motifs exists. Therefore, efficient local motif counting is a major challenge.

1.2. Our Contributions

We present several algorithms with time complexity

$$O\left(\frac{(3e)^k \cdot n \cdot |E| \cdot \log \frac{1}{\delta}}{\epsilon^2}\right)$$

that for the first time, approximate for every vertex the number of noninduced occurrences of the motif in which the vertex participates, for k -length cycles and

k -length cycles with a chord, where $k = O(\log n)$. We observe that while [Alon et al. 08] counts the total number of paths of length $O(\log n)$ in a graph, that technique is based on counting for each vertex the number of paths that start at the vertex and then summing for the entire network. Thus, their algorithm can be adapted for counting motifs adjacent to a node. For details see [Gonen and Shavitt 09]. We also provide algorithms with time complexity

$$O\left(\frac{n \cdot |E| \cdot \log \frac{1}{\delta}}{\epsilon^2} + |E|^2 \cdot \log n + |E| \cdot n \log n\right)$$

that for the first time, approximate for every vertex the number of noninduced occurrences of the motif in which the vertex participates for all motifs of size of at most 4. In addition, we provide

$$O\left(\frac{(2e)^k \cdot n \cdot |E| \cdot \log \frac{1}{\delta}}{\epsilon^2}\right)$$

algorithms that for the first time, approximate the total number of noninduced occurrences of $O(\log n)$ -length cycles with a chord. Moreover, we improve the time complexity of approximating the total number of noninduced occurrences of “tailed” triangles and 4-cliques over that of existing algorithms. Some of our algorithms use the “color-coding” technique of [Alon et al. 95] and techniques for using them [Alon et al. 08].

This paper is organized as follows: In Section 2 we give notation and definitions. In Section 3 we introduce motif-counting approximation algorithms for $O(\log n)$ -size motifs. In Section 4 we present motif-counting algorithms for all motifs of size 4. We summarize our conclusions in Section 5.

2. Preliminaries

Let $G = (V, E)$ be an undirected graph with n vertices. We assume that G is represented by an adjacency list. For a vertex v let $N(v)$ denote the set of neighbors of v and let $\deg(v)$ denote the degree of v . A motif H is said to be isomorphic to a subgraph H' in G if there is a bijection between the vertices of H and the vertices of H' such that for every edge between two vertices v and u of H there is an edge between the vertices v' and u' in H' that corresponds to v and u respectively. Such a subgraph H' is considered to be a noninduced occurrence of H in G . For a vertex v we say that v is *adjacent* to H if v is a vertex of H . Denote by $[k]$ the set $\{1, \dots, k\}$. Denote by $\text{col}(v)$ the color of vertex v .

Let H be a motif with k vertices, and let $G = (V, E)$ be a graph such that $|V| = n$. Assign a color to each vertex of V from the color set $[k]$. The colors are assigned to each vertex independently and uniformly at random. A copy of H in G is said to be *colorful* if each vertex on it is colored by a distinct color. Consider a problem f and let $\#f$ denote the number of distinct solutions of f .

Definition 2.1. ((ϵ, δ) -approximation.) An algorithm \mathcal{A} for a counting problem f is an (ϵ, δ) -approximation if it takes an input instance and two real values ϵ, δ and produces an output y such that

$$\Pr[(1 - \epsilon) \cdot \#f \leq y \leq (1 + \epsilon) \cdot \#f] \geq 1 - 2\delta.$$

3. Algorithms for Counting Motifs of size $O(\log n)$

Given a graph $G = (V, E)$ and a vertex v , we describe how to count for every vertex v the approximate number of noninduced occurrences of k -length cycles and k -length cycles with a chord that are adjacent to v , for $k = O(\log n)$. In addition, for each such motif H we present an algorithm for approximating the number of noninduced subgraphs of G that are isomorphic to H when no efficient algorithm exists. Most of our approximation algorithms apply the color-coding technique of [Alon et al. 95]. Note that we allow overlaps between the motifs we count, i.e., two occurrences of H , namely H' and H'' , may share vertices; in fact, the vertex sets of H' and H'' may be identical. We consider H' and H'' distinct occurrences of H if the edge sets of H' and H'' are not identical.

3.1. Counting Cycles

In this section assume that H is a simple cycle of length k :



We present an algorithm to count for every vertex v the approximate number of subgraphs of G that are isomorphic to H and adjacent to v :



Let $t = \log(1/\delta)$, and let $s = 4k^k/\epsilon^2 k!$. Assume that we have a k -coloring of G , i.e., each vertex is randomly and independently colored with a color in $[k]$. For each pair of vertices v, x and each color subset S of the color set $[k]$, let $C_i(v, x, S)$ be the number of colorful paths between v and x using colors in S at the i th coloring, and let $CY_i(v, S)$ be the number of colorful cycles adjacent to v using colors in S at the i th coloring.

Algorithm 1. (An (ϵ, δ) -approximation algorithm for counting simple cycles of length k adjacent to a vertex v)

1. For $j = 1$ to t
 - (a) For $i = 1$ to s
 - i. Color each vertex of G independently and uniformly at random with one of the k colors.
 - ii. For all $x \in V$, $C_i(v, x, [k]) = \text{count-path}(v, x, k)$. [See Algorithm 2.]
 - iii. Let $CY_i(v, [k]) = \frac{1}{2} \sum_{u \in N(v)} C_i(v, u, [k])$.
 - iv. Let $X_i^v = CY_i(v, [k])$.
 - (b) Let $Y_j^v = \frac{\sum_{i=1}^s X_i^v}{s}$.
 2. Let Z^v be the median of Y_1^v, \dots, Y_t^v .
 3. Return $Z^v \cdot k^k / k!$.
-

Consider Algorithm 1, which takes as input a graph $G = (V, E)$, a vertex $v \in V$, the requested cycle length k , an approximation factor ϵ , and an error probability δ . The algorithm uses a procedure to compute the number of colorful paths between v and any other vertex.

Our main theorem here is the following.

Theorem 3.1. *Let $G = (V, E)$ be an undirected graph, and let H be a simple cycle of length k . Then for every vertex v , Algorithm 1 is an (ϵ, δ) -approximation for the number of copies of H in G that are adjacent to v , with time complexity $O(((2e)^k \cdot n \cdot |E| \log(1/\delta)) / \epsilon^2)$.*

For proving Theorem 3.1 we first prove the following lemma.

Lemma 3.2. *For all $v \in V$, $CY_i(v, [k])$ can be computed in $O(2^k \cdot n \cdot |E|)$ time.*

Proof. A vertex v is adjacent to a colorful cycle of length k if and only if it is an endpoint of a colorful path of length $k - 1$ that has one of v 's neighbors as an endpoint:



Algorithm 2. (Procedure `count-path`(v, x, k) for counting simple paths of length $k - 1$ between v and x)

1. For all $S \subseteq [k]$ such that $S = \{\ell\}$, and for all $u \in V$,

$$C_i(u, x, S) = \begin{cases} 1 & \text{if } u = x \text{ and } \text{col}_i(u) = \ell; \\ 0 & \text{otherwise.} \end{cases}$$

2. For $q = 2$ to k

- (a) For all $S \subseteq [k]$ such that $|S| = q$,

$$C_i(v, x, S) = \sum_{u \in N(v)} C_i(u, x, S \setminus \{\text{col}_i(v)\}).$$

Therefore, we first compute for every edge $(u, v) \in E$ the number of colorful paths of length $k - 1$ between u and v . Since u is a neighbor of v , we get a cycle of length k . The running time for computing $CY(v, [k])$ for all v is then

$$O\left(\sum_{u \in V} \left(\sum_{v \in N(u)} \text{deg}(v)\right) 2^k\right) = O(2^k \cdot n \cdot |E|). \quad \square$$

Proof of Theorem 3.1. The correctness of the approximation returned by Algorithm 1 is proved using the same techniques as in [Alon et al. 08, Section 2]. Lemma 3.2 implies the correctness of the computation of $CY_i(v, [k])$. The time complexity of Algorithm 1 is $O((2e)^k \cdot n \cdot |E| \log(1/\delta)/\epsilon^2)$ by Lemma 3.2 and by showing that the number of colorings used by the algorithm is $O((e^k \log(1/\delta))/\epsilon^2)$. This completes the proof. \square

3.2. Counting k -Length Cycles with a Chord

In this subsection we assume that H is a simple cycle of length k with a chord:



We present an algorithm to compute the approximate number of subgraphs of G that are isomorphic to H , and for every vertex v , the number of subgraphs of G that are isomorphic to H and adjacent to v .

Algorithm 3. (An (ϵ, δ) -approximation algorithm for counting simple cycles of length k with a chord)

1. For $j = 1$ to t
 - (a) For $i = 1$ to s
 - i. Color each vertex of G independently and uniformly at random with one of the k colors.
 - ii. For all $(u, v) \in E$, $S \subseteq [k]$ compute $C_i(v, u, S)$
 - iii. Let $X_i = \sum_{(u,v) \in E} \sum_{\ell=2}^{k-2} \sum_{(S_1, S_2) \in A_{uv}^\ell} C_i(u, v, S_1) \cdot C_i(u, v, S_2)$.
 - (b) Let $Y_j = \frac{\sum_{i=1}^s X_i}{s}$.
 2. Let Z be the median of Y_1, \dots, Y_t .
 3. Return $Z \cdot k^k / k!$.
-

We first approximate the number of colorful subgraphs of G that are isomorphic to H , where H is a k -length cycle with a chord. Let $t = \log(1/\delta)$, let $s = \frac{4 \cdot k^k}{\epsilon^2 k!}$, and let count-path be the procedure defined in Algorithm 2. Assume that we have a k -coloring of G , i.e., each vertex is randomly and independently colored with a color in $[k]$. Let $C_i(v, u, S)$ be the number of colorful paths between v and u in the i th coloring, using the colors in S . Let ℓ be the distance between the endpoints of the chord u, v on the cycle, and let $A_{uv}^\ell = \{(S_1, S_2) \mid S_1, S_2 \subseteq [k], S_1 \setminus \{\text{col}(v), \text{col}(u)\} \cap S_2 \setminus \{\text{col}(v), \text{col}(u)\} = \emptyset, |S_1| = \ell + 1, |S_2| = k - \ell + 1\}$.

Consider Algorithm 3, which takes as input a graph $G = (V, E)$, an approximation factor ϵ , and an error probability δ .

Our main theorem here is the following.

Theorem 3.3. *Let $G = (V, E)$ be an undirected graph, and let H be a simple cycle of length k with a chord. Then Algorithm 3 is an (ϵ, δ) -approximation for the number of copies of H in G , with time complexity $O(|E| \cdot n \cdot (2e)^k \cdot \log(1/\delta)) / \epsilon^2$.*

For proving Theorem 3.3 we first prove the following lemma.

Lemma 3.4. *One can compute X_i with time complexity $O(|E| \cdot n \cdot 2^k)$.*

Proof. For computing X_i we compute for each edge $(u, v) \in E$ the number of colorful paths of length ℓ between u and v using a color set S_1 in the i th k -coloring $C_i(u, v, S_1)$ and the number of colorful paths of length $k - \ell$ between u and v using a color set S_2 in the i th k -coloring $C_i(u, v, S_2)$ such that $\text{col}(v)$ and $\text{col}(u)$ are the only colors in the intersection of S_1 and S_2 . The number of colorful subgraphs of G that are isomorphic to H that are counted in the i th k -coloring of G is then

$$\sum_{(u,v) \in E} \sum_{\ell=2}^{k-2} \sum_{(S_1, S_2) \in A_{uv}^\ell} C_i(u, v, S_1) \cdot C_i(u, v, S_2),$$

where $A_{uv}^\ell = \{(S_1, S_2) | S_1, S_2 \subseteq [k], S_1 \setminus \{\text{col}(v), \text{col}(u)\} \cap S_2 \setminus \{\text{col}(v), \text{col}(u)\} = \phi, |S_1| = \ell + 1, |S_2| = k - \ell + 1\}$.

As for the time complexity of computing X_i , by the proof of Lemma 3.2, the time complexity of computing $C_i(u, v, S)$ for every pair of vertices u, v and any color set S is $O(|E| \cdot n \cdot 2^k)$. In addition, we have to compute $|A_{uv}^\ell|$ for every $(u, v) \in E$ and every $2 \leq \ell \leq k - 2$. The running time for computing this operation is then $O(|E| \sum_{\ell=2}^{k-2} \binom{k}{\ell-1}) = O(|E| \cdot 2^k)$. Thus the total running time of computing X_i is $O(|E| \cdot n \cdot 2^k)$. \square

Proof of Theorem 3.3. Our proof of the correctness of the approximation returned by Algorithm 3 follows the technique of [Alon et al. 08]. Lemma 3.4 implies the correctness of the computation of X_i . The time complexity of Algorithm 3 is $O((|E| \cdot n \cdot (2e)^k \log(1/\delta))/\epsilon^2)$ by Lemma 3.4 and by showing that the number of colorings used by the algorithm is $O((e^k \log(1/\delta))/\epsilon^2)$. This completes the proof. \square

We now approximate for every $v \in V$ the number of colorful subgraphs of G that are isomorphic to H and adjacent to v . Let

$$t = \log(1/\delta) \quad \text{and} \quad s = \frac{4 \cdot k^k}{\epsilon^2 k!}.$$

Assume that we have a k -coloring of G , i.e., each vertex is randomly and independently colored with a color in $[k]$. Let $P_i(v, u, w, S)$ be the number of colorful paths from u to w that are adjacent to v in the i th coloring, using the colors in S . Recall that $C_i(v, u, S)$ is the number of colorful paths from v to u in the i th coloring, using the colors in S . Let $A_V^{z,b}(S) = \{(S_1, S_2) | S_1, S_2 \subseteq [k], |S_1| = z+1, |S_2| = b-z+1, S_1 \cup S_2 = S, S_1 \setminus \{\text{col}(u) | u \in V'\} \cap S_2 \setminus \{\text{col}(u) | u \in V'\} = \phi\}$. Consider Algorithm 4, which takes as input a graph $G = (V, E)$, a vertex v , an approximation factor ϵ , and an error probability δ .

Algorithm 4. (An (ϵ, δ) -approximation algorithm for counting simple cycles of length k with a chord that are adjacent to v)

1. For $j = 1$ to t
 - (a) For $i = 1$ to s
 - i. Color each vertex of G independently and uniformly at random with one of the k colors.
 - ii. $X_i^v = 0$.
 - iii. For every edge $(u, w) \in E$ and $S \subseteq [k]$ such that $|S| = \ell + 1$,

$$P_i(v, u, w, S) = \sum_{z=1}^{\ell-1} \sum_{(S_1, S_2) \in A_v^{z, \ell}(S)} C_i(v, w, S_1) \cdot C_i(v, u, S_2).$$
 - iv. Let

$$\begin{aligned} X_i^v &= X_i^v + \sum_{(u,v) \in E} \sum_{\ell=2}^{k-2} \sum_{(S_3, S_4) \in A_{uw}^{\ell, k}([k])} P_i(v, u, w, S_3) \cdot C_i(u, w, S_4) \\ &\quad + \sum_{(u,v) \in E} \sum_{\ell=2}^{k-2} \sum_{(S_3, S_4) \in A_{uw}^{k-\ell, k}([k])} P_i(v, u, w, S_3) \cdot C_i(u, w, S_4) \\ &\quad + \sum_{u \in N(v)} \sum_{\ell=2}^{k/2} \sum_{(S_3, S_4) \in A_{uv}^{\ell, k}([k])} C_i(v, u, S_3) \cdot C_i(v, u, S_4). \end{aligned}$$

- (b) Let $Y_j^v = \frac{\sum_{i=1}^s X_i^v}{s}$.

2. Let Z^v be the median of Y_1^v, \dots, Y_t^v .
 3. Return $Z^v \cdot k^k / k!$.
-

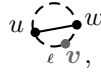
Our main theorem here is the following.

Theorem 3.5. *Let $G = (V, E)$ be an undirected graph, and let H be a simple cycle of length k with a chord. Then for every $v \in V$, Algorithm 4 is an (ϵ, δ) -approximation for the number of copies of H in G that are adjacent to v , with time complexity $O(|E| \cdot n \cdot (3e)^k \log(1/\delta) / \epsilon^2)$.*

For proving Theorem 3.5 we first prove the following lemma.

Lemma 3.6. *One can compute X_i^v with time complexity $O(|E| \cdot n \cdot e^k)$.*

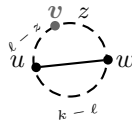
Proof. Let (u, w) be the chord. The number of copies of H that are adjacent to v depends on the position of v . There are two cases: one for which v is on a path between u and w and is not an endpoint of the chord:



and one for which v is an endpoint of the chord:



In the first case, we first count all the colorful paths of length ℓ between u and w of which v is part for all $2 \leq \ell \leq k-2$. We do that by counting all the colorful paths of length z between v and w and multiplying the result by the number of colorful paths of length $\ell - z$ between v and u , where $1 \leq z \leq \ell - 1$:



(We assume without loss of generality that $\ell \neq k - \ell$.)

Thus for all $S \subseteq [k]$ such that $|S| = \ell + 1$, we have

$$P_i(v, u, w, S) = \sum_{z=1}^{\ell-1} \sum_{(S_1, S_2) \in A_{v, \ell}^z(S)} C_i(v, w, S_1) \cdot C_i(v, u, S_2).$$

Therefore the total number of copies of H that are adjacent to v in the first case is the number of ℓ -length colorful paths between u and w that are adjacent to v , multiplied by the number of $(k - \ell)$ -length colorful paths between u and w with disjoint sets of colors (except for the colors of u and w):

$$\sum_{(S_3, S_4) \in A_{uw}^{\ell, k}([k])} P_i(v, u, w, S_3) \cdot C_i(u, w, S_4).$$

This should be computed for all $2 \leq \ell \leq k - 2$ and all $(u, w) \in E$. The second case is computed as follows. We count the number of ℓ -length colorful paths between u and v and multiply the result by the number of $(k - \ell)$ -length colorful paths between u and v , using disjoint sets of colors besides the colors of u and v . This is done for all $2 \leq \ell \leq k/2$. To compute the running time, according to the proof of Lemma 3.2, the time complexity for computing $C_i(v, w, S)$ for

every color set S and every pair of vertices v, w is $O(2^k \cdot n \cdot |E|)$. The running time of computing $P_i(v, u, w, S)$ for fixed vertices v, u, w , and every color set S (assuming that $C_i(v, w, S)$ is already computed) is

$$O\left(\sum_{\ell=1}^k \sum_{z=1}^{\ell-1} \binom{k}{\ell} \cdot \binom{\ell}{z}\right) = O\left(\sum_{\ell=1}^k \binom{k}{\ell} \cdot 2^\ell\right) = O(3^k).$$

Therefore the time complexity of computing the first case is

$$O\left(\sum_{v \in V} \sum_{(u,w) \in E} 3^k\right) + O(2^k \cdot n \cdot |E|) = O(3^k \cdot n \cdot |E|).$$

The time complexity of the second case (besides computing $C_i(v, w, S)$) is

$$O\left(\sum_{v \in V} \sum_{w \in N(v)} \sum_{\ell=1}^k \binom{k}{\ell}\right) = O(|E| \cdot 2^k).$$

Thus the total time complexity is $O(|E| \cdot n \cdot 3^k)$. \square

Proof of Theorem 3.5. The correctness of the approximation returned by Algorithm 4 is proved in the same manner as in the proof of Theorem 3.3. Lemma 3.6 implies the correctness of the computation of X_i^v . The time complexity of Algorithm 4 is $O((|E| \cdot n \cdot (3e)^k \log(1/\delta))/\epsilon^2)$ by Lemma 3.6 and by showing that the number of colorings used by the algorithm is $O((e^k \log(1/\delta))/\epsilon^2)$. This completes the proof. \square

4. Algorithms for Counting All Size-Four Motifs

Given a graph $G = (V, E)$ and a vertex v , we describe how to count for every vertex v the approximate number of noninduced occurrences of each possible motif H appearing in [Gordon et al. 10] that are adjacent to v . In addition, for each motif H that appears in [Gordon et al. 10] we present an algorithm for approximating the number of noninduced subgraphs of G that are isomorphic to H when no efficient algorithm exists. Note that we allow overlaps between the motifs, as in the previous section.

4.1. Counting “Tailed Triangles”

In this section assume that H is a triangle with a “tail” of length one:



We present an algorithm that approximates the number of subgraphs of G that are isomorphic to H and for every vertex v , approximates the number of subgraphs of G that are isomorphic to H and adjacent to v .

We first approximate the latter. There are three cases: one for which v is an endpoint of the path and adjacent to the triangle:



one for which v is not an endpoint of the path and adjacent to the triangle:



and one for which v is an endpoint of the path but not adjacent to the triangle:



Let $\text{TR}_G(v)$ be the approximation of the total number of triangles in G that are adjacent to v , according to Algorithm 1. Let $G_v = (V_v, E_v)$, where $V_v = V \setminus \{v\}$ and E_v is the induced set of edges obtained by removing all edges adjacent to v . Consider Algorithm 5, which takes as input a graph $G = (V, E)$, a vertex v , an approximation factor ϵ , and an error probability δ . Let $\text{TL}_G(v)$ be the number of “tailed triangles” in G returned by the algorithm.

Theorem 4.1. *Let $G = (V, E)$ be an undirected graph, and let H be a triangle with a “tail” of length one. Then for every vertex v , the number of copies of G that are isomorphic to H and adjacent to v can be (ϵ, δ) -approximated, with time complexity $O((n \cdot |E| \log(1/\delta))/\epsilon^2)$.*

Proof. In the first case



we get that the number of subgraphs of G that are isomorphic to H and adjacent to v is

$$\text{TR}_G(v) \cdot (|N(v)| - 2).$$

In the second case



Algorithm 5. (An (ϵ, δ) -approximation algorithm for counting simple “tailed triangles” adjacent to v)

1. $\text{TL}_G(v) = 0$.
 2. $\text{TR}_G(v) = \text{result of Algorithm 1 } (G, v, k = 3, \epsilon, \delta)$.
 3. $\text{TL}_G(v) = \text{TL}_G(v) + \text{TR}_G(v) \cdot (|N(v)| - 2)$.
 4. For all $u \in N(v)$:
 - (a) Compute $N(v) \cap N(u)$:
 - i. Go over all the vertices in the adjacency list of v and the adjacency list of u , and add each vertex to a list. (Thus a vertex can appear several times in the list.)
 - ii. For each vertex in the list count the number of times it appears in the list. If it appears twice, then add the vertex to a list $\ell(u, v)$.
 - (b) For all $w \in \ell(u, v)$ $\text{TL}_G(v) = \text{TL}_G(v) + \deg w - 2 + \deg u - 2$.
 5. Compute G_v by going over the whole adjacency list and removing v any time it appears in the list.
 6. For all $u \in N(v)$, $\text{TR}_{G_v}(u) = \text{result of Algorithm 1 } (G_v, u, k = 3, \epsilon, \delta)$.
 7. $\text{TL}_G(v) = \text{TL}_G(v) + \sum_{u \in N(v)} \text{TR}_{G_v}(u)$.
 8. Return $\text{TL}_G(v)$.
-

we get that the number of subgraphs of G that are isomorphic to H and adjacent to v is

$$\sum_{u \in N(v)} \sum_{w \in N(v) \cap N(u)} (\deg w - 2 + \deg u - 2).$$

In the third case



we get that the number of subgraphs of G that are isomorphic to H and adjacent to v is

$$\sum_{u \in N(v)} \text{TR}_{G_v}(u).$$

Thus the total number of subgraphs of G that are isomorphic to H and adjacent to v is

$$\text{TR}_G(v) \cdot (|N(v)| - 2) + \sum_{u \in N(v)} \sum_{w \in N(v) \cap N(u)} (\deg w - 2 + \deg u - 2) + \sum_{u \in N(v)} \text{TR}_{G_v}(u).$$

Let \hat{r}_v be the approximated value for the number of subgraphs of G that are isomorphic to H and adjacent to v in the first and third cases. Let r_v be the exact number. In a similar manner to that of Theorem 3.1, \hat{r}_v is an (ϵ, δ) -approximation to the number of subgraphs of G that are isomorphic to H and adjacent to v in the first and third cases. For the second case, Algorithm 5 gives an exact solution. Let a_v be the number of subgraphs of G that are isomorphic to H and adjacent to v contributed by this case. We need to show that $\hat{r}_v + a_v$ is an (ϵ, δ) -approximation for the total number of subgraphs of G that are isomorphic to H and are adjacent to v :

$$\begin{aligned} & \Pr[\hat{r}_v + a_v \in [(1 - \epsilon)(r_v + a_v), (1 + \epsilon)(r_v + a_v)]] \\ & \geq \Pr[\hat{r}_v + a_v \in [(1 - \epsilon)r_v + a_v, (1 + \epsilon)r_v + a_v]] \\ & \geq 1 - 2\delta. \end{aligned}$$

This completes the proof of the correctness of the algorithm.

Assuming that the degree of every vertex is known, the time complexity of finding the total number of subgraphs of G that are isomorphic to H and adjacent to v is the time of computing the number of triangles that are adjacent to v , for every v , plus the time of computing G_v for every vertex v , plus the time of computing $N(v) \cap N(u)$ for every $u \in N(v)$, for every vertex v , plus the time of computing the number of triangles that are adjacent to u for every vertex $u \in N(v)$, for every vertex v . By Theorem 3.1, the time complexity of computing the number of triangles that are adjacent to v for every v is $O((n \cdot |E| \log(1/\delta))/\epsilon^2)$. The time complexity of computing G_v for every vertex v is $O(n \cdot |E|)$. The time complexity for computing $N(v) \cap N(u)$ for every $u \in N(v)$, for every $v \in V$, is

$$\begin{aligned} O\left(\sum_{v \in V} \sum_{u \in N(v)} \deg u + \deg v\right) &= O\left(\sum_{v \in V} \sum_{u \in N(v)} n\right) \\ &= O\left(n \cdot \sum_{v \in V} \deg v\right) = O(|E| \cdot n). \end{aligned}$$

Thus the total time complexity is

$$O\left(\frac{n \cdot |E| \log(1/\delta)}{\epsilon^2}\right) + O(n \cdot |E|) = O\left(\frac{n \cdot |E| \log(1/\delta)}{\epsilon^2}\right). \quad \square$$

We now count the number of subgraphs of G that are isomorphic to H , where H is a tailed triangle. According to the above, the number of subgraphs of G that are isomorphic to H is

$$\sum_{v \in V} \text{TR}_G(v) \cdot (|N(v)| - 2).$$

Assuming that for every v , $\deg v$ is known, by Theorem 4.1 the time complexity is $O((n \cdot |E| \log(1/\delta))/\epsilon^2)$. Therefore we immediately get the following theorem.

Theorem 4.2. *Let $G = (V, E)$ be an undirected graph, and let H be a triangle with a “tail” of length 1. Then the number of copies of G that are isomorphic to H can be (ϵ, δ) -approximated, with time complexity $O((n \cdot |E| \log(1/\delta))/\epsilon^2)$.*

4.2. Counting 4-Cliques

In this section we assume that H is a clique of size four:



We present an algorithm that *exactly* computes the number of subgraphs of G that are isomorphic to H and for every vertex v , the number of subgraphs of G that are isomorphic to H and adjacent to v .

We first compute, for every vertex v , the number of subgraphs of G that are isomorphic to H and adjacent to v :



Let $\text{Cl}(v)$ be the number of four-cliques in the graph that are adjacent to v . Algorithm 6 takes as input a graph $G = (V, E)$ and a vertex v .

Theorem 4.3. *Let $G = (V, E)$ be an undirected graph, and let H be a clique of size four. Then for all $v \in V$, Algorithm 6 counts the number of copies of H in G that are adjacent to v , with time complexity $O(|E| \cdot n \log n + |E|^2 \cdot \log n)$.*

Proof. The correctness of Algorithm 6 is trivial. Using a computation similar to the one in the proof of Theorem 4.1, we get the following for the time complexity of

Algorithm 6. (Algorithm for counting 4-cliques that are adjacent to v)

1. $\text{Cl}(v) = 0$.
 2. For every vertex $u \in N(v)$:
 - (a) Compute $N(v) \cap N(u)$:
 - i. Go over all the vertices in the adjacency list of v and the adjacency list of u , and add each vertex to a list. (Thus a vertex can appear several times in the list.)
 - ii. For each vertex in the list count the number of times it appears in the list. If it appears twice then add the vertex to a list $\ell(u, v)$.
 - iii. Sort the list $\ell(u, v)$ according to the names of the vertices.
 - (b) For all $w \in \ell(u, v)$ go over the adjacency list of w and for each vertex $t \neq v, u$ in this adjacency list check whether $t \in \ell(u, v)$. If $t \in \ell(u, v)$ then $\text{Cl}(v) := \text{Cl}(v) + 1$.
 3. Return $\text{Cl}(v)/6$.
-

Algorithm 6:

$$\begin{aligned}
 & O\left(\sum_{v \in V} \sum_{u \in N(v)} \deg u + \deg v\right) \\
 & + O\left(\sum_{v \in V} \sum_{u \in N(v)} |N(v) \cap N(u)| \log(|N(v) \cap N(u)|)\right. \\
 & \quad \left. + \sum_{w \in N(v) \cap N(u)} \deg w \cdot \log(|N(v) \cap N(u)|)\right) \\
 & = O(|E| \cdot n \log n) + O\left(\sum_{v \in V} \sum_{u \in N(v)} \sum_{w \in N(v) \cap N(u)} \deg w \cdot \log(|N(v) \cap N(u)|)\right) \\
 & = O(|E| \cdot n \log n) + O\left(\sum_{v \in V} \sum_{u \in N(v)} \sum_{w \in V} \deg w \cdot \log n\right) \\
 & = O(|E| \cdot n \log n) + O(|E|^2 \cdot \log n). \quad \square
 \end{aligned}$$

We now count the number of subgraphs of G that are isomorphic to H , where H is a four-clique. Let Cl be the total number of four-cliques in the graph. Then computing Cl immediately follows by the previous algorithm:

$$\text{Cl} = \frac{1}{4} \sum_{v \in V} \text{Cl}(v).$$

Theorem 4.4. *Let $G = (V, E)$ be an undirected graph, and let H be a clique of size four. Then the number of copies of H in G can be computed with time complexity $O(|E| \cdot n \log n + |E|^2 \cdot \log n)$.*

4.3. Counting Small Trees

In this section assume that H is a tree of size four that consists of a vertex and three of its neighbors. We present an algorithm that exactly computes, for every vertex v , the number of subgraphs of G that are isomorphic to H and adjacent to v . There are two cases: one for which v is an endpoint of all edges of the tree:



and one for which v is an endpoint of only one edge:



In the first case we get $\binom{|N(v)|}{3}$. In the second case, for every $u \in N(v)$ we count all subsets of size 2 of $N(u)$ (not including v). Therefore this case contributes $\sum_{u \in N(v)} \binom{\deg(u)-1}{2}$ subgraphs of G that are isomorphic to H and are adjacent to v . Thus the total number of subgraphs of G that are isomorphic to H and adjacent to v is

$$\binom{\deg(v)}{3} + \sum_{u \in N(v)} \binom{\deg(u)-1}{2}.$$

Assuming that the degree of every vertex is known, the time complexity is

$$\sum_{v \in V} (O(1) + O(\deg(v))) = O(|E| + n).$$

Thus we immediately get the following theorem.

Theorem 4.5. *Let $G = (V, E)$ be an undirected graph, and let H be a tree of size four that consists of a vertex and three of its neighbors. Then for every vertex v , the number of subgraphs of G that are isomorphic to H and adjacent to v can be found with time complexity $O(|E| + n)$.*

Note that the total number of subgraphs of G that are isomorphic to H can be easily counted using the first case, for all v , with time complexity $O(n)$.

5. Conclusions

In this work we have presented algorithms with time complexity

$$O\left(\frac{(3e)^k \cdot n \cdot |E| \cdot \log \frac{1}{\delta}}{\epsilon^2}\right)$$

that for the first time, approximate for every vertex the number of noninduced occurrences of the motif of which the vertex is part, for k -length cycles and k -length cycles with a chord, where $k = O(\log n)$. We also designed algorithms with time complexity

$$O\left(\frac{n \cdot |E| \cdot \log \frac{1}{\delta}}{\epsilon^2} + |E|^2 \cdot \log n + |E| \cdot n \log n\right)$$

that for the first time, approximate for every vertex the number of noninduced occurrences of the motif of which the vertex is part, for all motifs of size at most four. In addition, we have given algorithms that approximate the total number of noninduced occurrences of these network motifs when no efficient algorithm exists. Approximating the number of noninduced occurrences of the motif of which a vertex is part for other motifs of size $O(\log n)$ is left for future work.

Acknowledgments. We thank Dana Ron for many hours of fruitful discussions, and the anonymous reviewers for their help in improving this manuscript. This work was partially funded by the OneLab II consortium, which is partly financed by the European Commission, and by the Israeli Science Foundation center of knowledge on communication networks (grant #1685/07). This research was conducted while the first author was at the School of Electrical Engineering at Tel-Aviv University.

References

- [Albert and Barabási 00] R. Albert and A.-L. Barabási. “Topology of Evolving Networks: Local Events and Universality.” *Physical Review Letters* 85:24 (2000), 5234–5237.
- [Alon and Gutner 07] N. Alon and S. Gutner. “Balanced Families of Perfect Hash Functions and Their Applications.” In *Automata, Languages and Programming: 34th International Colloquium, ICALP 2007, Wrocław, Poland, July 9–13, 2007, Proceedings*, Lecture Notes in Computer Science 4596, pp. 435–446. Berlin: Springer, 2007.

- [Alon et al. 95] N. Alon, R. Yuster, and U. Zwick. “Color Coding.” *Journal of the ACM* 42 (1995), 844–856.
- [Alon et al. 97] N. Alon, R. Yuster, and U. Zwick. “Finding and Counting Given Length Cycles.” *Algorithmica* 17 (1997), 209–223.
- [Alon et al. 08] N. Alon, P. Dao, I. Hajirasouliha, F. Hormozdiari, and S. C. Sahinalp. “Biomolecular Network Motif Counting and Discovery by Color Coding.” *Bioinformatics* 1 (2008), 1–9.
- [Arvind and Raman 02] V. Arvind and V. Raman. “Approximation Algorithms for Some Parameterized Counting Problems.” In *Algorithms and Computation: 13th International Symposium, ISAAC 2002 Vancouver, BC, Canada, November 21–23, 2002, Proceedings*, Lecture Notes in Computer Science 2518, pp. 453–464. Berlin: Springer, 2002.
- [Bar et al. 04] S. Bar, M. Gonen, and A. Wool. “An Incremental Super-linear Preferential Internet Topology Model.” In *Passive and Active Network Measurement: 5th International Workshop, PAM 2004, Antibes Juan-les-Pins, France, April 19–20, 2004, Proceedings*, Lecture Notes in Computer Science 3015, pp. 53–62. Berlin: Springer, 2004.
- [Bar et al. 05] S. Bar, M. Gonen, and A. Wool. “A Geographic Directed Preferential Internet Topology Model. In *Proceedings of the 13th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, pp. 325–328. Washington, DC: IEEE Computer Society, 2005.
- [Barford et al. 01] P. Barford, A. Bestavros, J. Byers, and M. Crovella. “On the Marginal Utility of Network Topology Measurements.” In *Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement*, pp. 5–17. New York: ACM, 2001.
- [Becchetti et al. 08] L. Becchetti, P. Boldi, C. Castillo, and A. Gionis. “Efficient Semi-streaming Algorithms for Local Triangle Counting in Massive Graphs.” In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 16–24. New York: ACM, 2008.
- [Bianconi and Barabási 01] G. Bianconi and A. L. Barabási. “Competition and Multiscaling in Evolving Networks.” *Europhysics Letters* 54:4 (2001), 436–442.
- [Bianconi and Capocci 03] G. Bianconi and A. Capocci. “Number of Loops of Size h in Growing Scale-Free Networks.” *Physical Review Letters* 90 (2003), 078701.
- [Bianconi and Marsili 05] G. Bianconi and M. Marsili. “Loops of Any Size and Hamilton Cycles in Random Scale-Free Networks.” *Journal of Statistical Mechanics* (2005), P06005.
- [Bianconi and Marsili 06] G. Bianconi and M. Marsili. “Number of Cliques in Random Scale-Free Network Ensembles.” *Physica D* 224 (2006), 1–6.
- [Brunet and Sokolov 02] R. X. Brunet and I. M. Sokolov. “Evolving Networks with Disadvantaged Long-Range Connections.” *Physical Review E* 66 (2002), 026118.
- [Bu and Towsley 02] T. Bu and D. Towsley. “On Distinguishing between Internet Power-Law Topology Generators.” In *Proceedings of the Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*, pp. 638–647. Washington, DC: IEEE Computer Society, 2002.

- [Chen et al. 02] Q. Chen, H. Chang, R. Govindan, S. Jamin, S. Shenker, and W. Willinger. “The Origin of Power Laws in Internet Topologies Revisited.” In *Proceedings of the Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*, pp. 608–617. Washington, DC: IEEE Computer Society, 2002.
- [Dost et al. 07] B. Dost, T. Shlomi, N. Gupta, E. Ruppim, V. Bafna, and R. Sharan. “QNet: A Tool for Querying Protein Interaction Networks.” In *Research in Computational Molecular Biology: 11th Annual International Conference, RECOMB 2007, Oakland, CA, USA, April 21–25, 2007, Proceedings*, Lecture Notes in Computer Science 4453, pp. 1–15. New York: Springer, 2007.
- [Duke et al. 95] R. Duke, H. Lefmann, and V. Rödl. “A Fast Approximation Algorithm for Computing the Frequencies of Subgraphs in a Given Graph.” *SIAM Journal on Computing* 24:3 (1995), 598–620.
- [Faloutsos et al. 99] C. Faloutsos, M. Faloutsos, and P. Faloutsos. “On Power-Law Relationships of the Internet Topology.” In *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pp. 251–260. New York: ACM, 1999.
- [Feldman and Shavitt 08] Dima Feldman and Yuval Shavitt. “Automatic Large Scale Generation of Internet PoP Level Maps.” In *Proceedings of the IEEE Global Telecommunications Conference*, pp. 1–6. Washington, DC: IEEE Computer Society, 2008.
- [Gonen and Shavitt 09] Mira Gonen and Yuval Shavitt. “Approximating the Number of Network Motifs.” In *Algorithms and Models for the Web-Graph: 6th International Workshop, WAW 2009, Barcelona, Spain, February 12–13, 2009, Proceedings*, Lecture Notes in Computer Science 5427, pp. 13–24. New York: Springer, 2009.
- [Gonen et al. 10] M. Gonen, D. Ron, and Y. Shavitt. “Counting Stars and Other Small Subgraphs in Sublinear Time.” In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 99–116. Philadelphia: SIAM, 2010.
- [Gordon et al. 10] M. Gordon, E. A. Livneh, R. Y. Pinter, and E. Rubin. “Elucidating Protein Function Using Graphlet Degree Vectors in Protein–Protein Interactions Networks.” To appear, 2010.
- [Govindan and Tangmunarunki 07] R. Govindan and H. Tangmunarunki. “Heuristics for Internet Map Discovery.” In *Proceedings of the Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies*, pp. 1371–1380. Washington, DC: IEEE Computer Society, 2000.
- [Grochow and Kellis 07] J. Grochow and M. Kellis. “Network Motif Discovery Using Subgraph Enumeration and Symmetry-Breaking.” In *Research in Computational Molecular Biology: 11th Annual International Conference, RECOMB 2007, Oakland, CA, USA, April 21–25, 2007, Proceedings*, Lecture Notes in Computer Science 4453, pp. 92–106. New York: Springer, 2007.
- [Hales and Arteconi 08] D. Hales and S. Arteconi. “Motifs in Evolving Cooperative Networks Look like Protein Structure Networks.” *Special Issue of ECCS’07 in the Journal of Networks and Heterogeneous Media* 3:2 (2008), 239–249.

- [Hormozdiari et al. 07] F. Hormozdiari, P. Berenbrink, N. Przulj, and S. C. Sahinalp. “Not All Scale-Free Networks Are Born Equal: The Role of the Seed Graph in PPI Network Evolution.” *PLoS: Computational Biology* 3:7 (2007), e118.
- [Itzhack et al. 07] R. Itzhack, Y. Mogilevski, and Y. Louzoun. “An Optimal Algorithm for Counting Network Motifs.” *Physica A* 381 (2007), 482–490.
- [Karp and Luby 83] R. Karp and M. Luby. “Monte-Carlo Algorithms for Enumeration and Reliability Problems.” In *Proceedings of the Twenty-Fourth Annual Symposium on Foundations of Computer Science*, pp. 56–64. Washington, DC: IEEE Computer Society, 1983.
- [Kashtan et al. 04] N. Kashtan, S. Itzkovitz, R. Milo, and U. Alon. “Efficient Sampling Algorithm for Estimating Subgraph Concentrations and Detecting Network Motifs.” *Bioinformatics* 20:11 (2004), 1746–1758.
- [Lakhina et al. 03] A. Lakhina, J. W. Byers, M. Crovella, and P. Xie. “Sampling Biases in IP Topology Measurements.” In *Proceedings of the Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies*, pp. 332–341. Washington, DC: IEEE Computer Society, 2003.
- [Li and Chen 03] X. Li and G. Chen. “A Local-World Evolving Network Model.” *Physica A* 328 (2003), 274–286.
- [Marinari et al. 07] E. Marinari, G. Semerjian, and V. Van Kerrebroeck. “Finding Long Cycles in Graphs.” *Physical Review E* 75:6 (2007), 066708.
- [Milo et al. 02] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. “Network Motifs: Simple Building Blocks of Complex Networks.” *Science* 298 (2002), 824–827.
- [Przulj 07] N. Przulj. “Biological Network Comparison Using Graphlet Degree Distribution.” *Bioinformatics* 23:2 (2007), e177–e183.
- [Przulj et al. 05] N. Przulj, D. G. Corneil, and I. Jurisica. “Modeling Interactome: Scale-Free or Geometric?” *Bioinformatics* 20:18 (2004), 3508–3515.
- [Reittu and Norros 04] H. Reittu and I. Norros. “On the Power Law Random Graph Model of the Internet.” *Performance Evaluation* 55, 2004.
- [Sagie and Wool 04] G. Sagie and A. Wool. “A Clustering Approach for Exploring the Internet Structure.” In *Proceedings of the 23rd IEEE Convention of Electrical and Electronics Engineers in Israel*, pp. 149–152. Washington, DC: IEEE Computer Society, 2004.
- [Scott et al. 05] J. Scott, T. Ideker, R. Karp, and R. Sharan. “Efficient Algorithms for Detecting Signaling Pathways in Protein Interaction Networks.” In *Research in Computational Molecular Biology: 9th Annual International Conference, RECOMB 2005, Cambridge, MA, USA, May 14–18, 2005, Proceedings*, Lecture Notes in Computer Science 3500, pp. 1–13. New York: Springer, 2005.
- [Shavitt and Shir 05] Yuval Shavitt and Eran Shir. “DIMES: Let the Internet Measure Itself.” *ACM SIGCOMM Computer Communication Review* 35 (2005), 71–74.
- [Shlomi et al. 06] T. Shlomi, D. Segal, and E. Ruppin. “QPath: A Method for Querying Pathways in a Protein–Protein Interaction Network.” *Bioinformatics* 7 (2006), 199.

- [Siganos et al. 06] G. Siganos, S. L. Tauro, and M. Faloutsos. “Jellyfish: A Conceptual Model for the AS Internet Topology.” *Journal of Communications and Networks* 8 (2006), 339–350.
- [Subramanian et al. 02] L. Subramanian, S. Agarwal, J. Rexford, and R. H. Katz. “Characterizing the Internet Hierarchy from Multiple Vantage Points.” In *Proceedings of the Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*, pp. 618–627. Washington, DC: IEEE Computer Society, 2002.
- [Tangmunarunkit et al. 02] H. Tangmunarunkit, R. Govindan, S. Jamin, S. Shenker, and W. Willinger. “Network Topology Generators: Degree Based vs. Structural.” In *Proceedings of the 2002 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pp. 147–159. New York: ACM, 2002.
- [Tauro et al. 01] L. Tauro, C. Palmer, G. Siganos, and M. Faloutsos. “A Simple Conceptual Model for the Internet Topology.” In *Proceedings of the IEEE Conference on Global Telecommunications*, pp. 1667–1671. Washington, DC: IEEE Computer Society, 2001.
- [Van Kerrebroeck and Marinari 08] V. Van Kerrebroeck and E. Marinari. “Ranking by Loops: A New Approach to Categorization.” *Physical Review Letters* 101 (2008), 098701.
- [Wernicke 06] S. Wernicke. “Efficient Detection of Network Motifs.” *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 3:4 (2006), 347–359.
- [Willinger et al. 02] W. Willinger, R. Govindan, S. Jamin, V. Paxson, and S. Shenker. “Scaling Phenomena in the Internet: Critically Examining Criticality.” *Proceedings of the National Academy of Sciences of the United States of America* 99 (2002), 2573–2580.
- [Winick and Jamin 02] J. Winick and S. Jamin. “INET-3.0: Internet Topology Generator.” Technical Report UM-CSE-TR-456-02, Department of EECS, University of Michigan, 2002.

Mira Gonen, Mathematics Department, Bar-Ilan University, Ramat Gan, Israel
(gonem1@math.biu.ac.il)

Dana Ron, School of Electrical Engineering, Tel-Aviv University, Ramat Aviv, Israel
(danar@eng.tau.ac.il)

Yuval Shavitt, School of Electrical Engineering, Tel-Aviv University, Ramat Aviv, Israel
(shavitt@eng.tau.ac.il)

Received June 29, 2009; accepted June 3, 2010.