

# Approximating the number of Network Motifs

Mira Gonen<sup>1\*</sup> and Yuval Shavitt<sup>2\*\*</sup>

<sup>1</sup> Bar Ilan University, Ramat Gan, Israel

<sup>2</sup> Tel-Aviv University, Ramat Aviv, Israel

**Abstract.** World Wide Web, the Internet, coupled biological and chemical systems, neural networks, and social interacting species, are only a few examples of systems composed by a large number of highly interconnected dynamical units. These networks contain characteristic patterns, termed *network motifs*, which 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 in the form of trees and bounded treewidth subgraphs of size  $O(\log n)$ , and of size at most 7 for some motifs.

In addition, *local motif counting*, namely, counting the number of motifs a node is part of, was recently suggested as a method to classify nodes in the network. The promise is that the distribution of motifs a node participate in is an indication of its function in the network. Therefore, local counting network motifs provides a major challenge. However, no such practical algorithm exists, besides for local counting of triangles.

We present several algorithms with time complexity  $O\left(\frac{(3e)^k \cdot n \cdot |E| \cdot \log \frac{1}{\epsilon}}{\epsilon^2}\right)$  that, for the first time, approximate for every vertex the number of occurrences of the motif the vertex is part of, for  $k$ -length cycles and  $k$ -length cycles with a chord, where  $k = O(\log n)$ , and algorithms with time complexity  $O\left(\frac{n \cdot |E| \cdot \log \frac{1}{\epsilon}}{\epsilon^2} + |E|^2 + |E| \cdot n \log n\right)$  that approximate for every vertex the number of non-induced occurrences of the motif the vertex is part of 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.

## 1 Introduction

### 1.1 Background and Motivation

World Wide Web, the Internet, coupled biological and chemical systems, neural networks, and social interacting species, are only a few examples of systems composed by a large number of highly interconnected dynamical units. The first approach to capture the global properties of such systems is to model them as

---

\* Email: [gonenm1@math.biu.ac.il](mailto:gonenm1@math.biu.ac.il). This work was done while the author was at Tel Aviv University.

\*\* Email: [shavitt@eng.tau.ac.il](mailto:shavitt@eng.tau.ac.il).

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 dense-core and small diameter [20, 1, 17, 31, 47, 48, 16, 8, 7, 21, 40, 32, 11, 9, 37, 45, 43, 44, 23, 15, 38, 42]. However, two networks which have similar global features, such as similar degree sequence, can have significant differences in structure, which can be captured by examining local structures they include: e.g., one of them may include a specific subgraph many more times than the other. Therefore these small subgraphs, termed network motifs, were suggested to be elementary building blocks that carry out key functions in the network. Milo *et al.* [34] found motifs in networks from biochemistry, neurobiology, ecology, and the World Wide Web. Moreover, Hales and Arteconi [25] presented results from a motif analysis of networks produced by peer-to-peer protocols. They showed 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 of counting their number. One of the most elegant techniques devised is *color-coding*, introduced in [4], and further applied in [5, 6, 3, 2, 18, 41]. 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 where each vertex 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 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 [6] used color-coding and a technique from [28] to design a randomized algorithm for approximately counting the number of subgraphs in a given graph  $G$  which 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 number of vertices in  $G$  and  $H$ , respectively, and  $b$  is the treewidth of  $H$ . Alon and Gutner [3] used 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  $2^{O(k \log \log k)} n^{O(1)}$ . Alon *et al.* [2] improved these results in terms of the dependence on  $k$ .

Przulj *et al.* [36] described how to *count* all *induced*<sup>1</sup> subgraphs with up to 5 vertices in a PPI (Protein-Protein Interaction) network. Hormozdiari *et al.* [26] developed faster techniques for counting induced subgraphs of size up to

---

<sup>1</sup> Note 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  $G$ .

6, and Grochow and Kellis [24] developed such techniques for size up to 7. The running time of these techniques all increase exponentially with the size of the motif. Kashtan *et al.* [29] showed an algorithm for *detecting* induced network motifs that sample the network. This algorithm detect induced occurrences of small motifs (motifs with  $k \leq 7$  vertices). Wernicke *et al.* [46] claimed that Kashtan *et al.*'s algorithm suffers from a sampling bias and scales poorly with increasing subgraph size. Thus, Wernicke [46] presented an improved algorithm for network motif detection which overcomes these drawbacks. Scott *et al.* [39] focused on the subgraph detection problem. Dost *et al.* [18] showed how to solve the subgraph detection problem for subgraphs of size  $O(\log n)$ , provided that the query subgraph is either a simple path, a tree, or a bounded treewidth subgraph. Duke *et al.* [19] gave an algorithm which provides the number of induced copies of certain subgraphs, with a bounded error. Their algorithm has running time of  $O(n^{1/\log \log n} \cdot M(n))$ , where  $M(n)$  is the time needed to square an  $n$  by  $n$  matrix with 0, 1-entries over the integers. Itzhack *et al.* [27] presented an algorithm based on network decomposition via node removal for counting  $k$ -size network motifs in large networks, where  $k$  is 3 or 4. Their algorithm detects all motifs containing a given node by measuring all its incoming and outgoing neighbors of degree  $k - 1$ , and then remove this node. Their 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 [12] showed 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 [13] evaluated the average number of cycles in random scale-free networks. They showed that the most frequent size of a cycle in a scale-free network is of the order of the network's size. Moreover, they indicated that small length cycles are more frequent when the second moment of the degree distribution diverges. Marinari *et al.* [33] analyzed the problem of discovering long cycles in random graphs. They proposed and tested two algorithms for this task. The first one is based on a message passing procedure, and the second follows a Monte Carlo Markov chain strategy. Bianconi and Marsili [14] counted the average number of cliques in random scale-free networks when the network is large. They showed that unlike in *Erdős* Renyi graphs, cliques appear also when the average degree is finite. In addition, the authors proved that in random scale-free networks the clique number diverges with the system size.

Gonen *et al.* [?] introduced a *sublinear* algorithm for approximating the number of constant size stars. Their algorithm is based on querying parts of the graph. They showed that their result is tight up to polylogarithmic factors in  $n$  and the dependence in  $\epsilon$  (the counting error) by giving a matching lower bound. They also showed negative results for sublinear counting of length-3 paths and triangles.

A new systematic measure of a network's *local* topology was recently suggested by Przulj [35]. They term this measure "graphlet distribution" of a vertex. Namely, they count for each vertex the number of all motifs of size at most five the vertex is part of. Gordon *et al.* [22] discussed local motif counting as a

method to classify nodes in the network. The promise is that the distribution of motifs a node participate in is an indication of its function in the network, thus nodes can be divided into functional classes. Van Kerrebroeck and Marinari [30] suggested the number of cycles a vertex participate in as a method to quantify the role of vertices in the network. Becchetti *et al.* [10] have recently shown 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  $O\left(\frac{(3\epsilon)^k \cdot n \cdot |E| \cdot \log \frac{1}{\epsilon}}{\epsilon^2}\right)$  that, for the first time, approximate for every vertex the number of non-induced occurrences of the motif the vertex is part of, for  $k$ -length cycles,  $k$ -length cycles with a chord, where  $k = O(\log n)$ . We observe that while Alon *et al.* [2] counted the total number of paths of length  $O(\log n)$  in a graph, their technique is based on counting for each vertex the number of paths that start at the vertex and then summing up for the entire network. Thus, their algorithm can be adapted for counting motifs adjacent to a node. For details see [?]. We also provide algorithms with time complexity  $O\left(\frac{n \cdot |E| \cdot \log \frac{1}{\epsilon}}{\epsilon^2} + |E|^2 + |E| \cdot n \log n\right)$  that, for the first time, approximate for every vertex the number of non-induced occurrences of the motif the vertex is part of for all motifs of size of at most four. In addition, we show  $O\left(\frac{(2\epsilon)^k \cdot n \cdot |E| \cdot \log \frac{1}{\epsilon}}{\epsilon^2}\right)$  algorithms that, for the first time, approximate the total number of non-induced occurrences of  $O(\log n)$ -length cycles with a chord. Moreover, we improve the time complexity of approximating the total number of non-induced occurrences of “tailed” triangles and 4-cliques upon existing algorithms. Some of our algorithms use the color coding technique of Alon *et al.* [4], and the techniques to use them [2].

*Organization:* In Section 2 we give notations and definitions. In Section 3 we introduce motifs counting approximation algorithms for  $O(\log n)$ -size motifs. In Section 4 we present motifs counting algorithms for all four-size motifs. 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 correspond to  $v$  and  $u$  respectively. Such a subgraph  $H'$  is considered to be a non-induced 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  $col(v)$  the color of vertex  $v$ .

Let  $H$  be a motif with  $k$  vertices, and let  $G = (V, E)$  be a graph where  $|V| = n$ . 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 1. ( $(\epsilon, \delta)$ -approximation)** An algorithm  $A$  for a counting problem  $f$  is a  $(\epsilon, \delta)$ -approximation if it takes an input instance and two real values  $\epsilon, \delta$  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 approximately count for every vertex  $v$  the number of non-induced 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 non-induced 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.* [4]. 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  $H$  provided that 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 approximately count for every vertex  $v$  the number of subgraphs of  $G$  which are isomorphic to  $H$  and adjacent to  $v$ . 

Let  $t = \log(1/\delta)$ , and let  $s = \frac{4k^k}{\epsilon^2 k!}$ . 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.

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

**Algorithm 1** (A  $(\epsilon, \delta)$ -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  $x \in V$   $C_i(v, x, [k]) = \text{count-path}(v, x, k)$ .
    - 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  $Z^v \cdot k^k / k!$ .

**Algorithm 2**  $\text{count-path}(v, x, k)$  (counting simple paths of length  $k - 1$  between  $v$  and  $x$ )

1. For all  $S \subseteq [k]$  s.t.  $S = \{\ell\}$ , 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  $S \subseteq [k]$  s.t.  $|S| = q$


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

Our main theorem is the following:

**Theorem 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 a  $(\epsilon, \delta)$ -approximation for the number of copies of  $H$  in  $G$  that are adjacent to  $v$ , with time complexity  $O\left(\frac{(2e)^k \cdot n \cdot |E| \log(1/\delta)}{\epsilon^2}\right)$ .

For proving Theorem 1 we first prove the following lemma:

**Lemma 1** 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  $k - 1$ , which has one of  $v$ 's neighbor as an endpoint.  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

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

□

**Proof of Theorem 1.** The correctness of the approximation returned by Algorithm 1 is proved using the same techniques as in [2, Sec. 2]. Lemma 1 implies the correctness of the computation of  $CY_i(v, [k])$ . The time complexity of Algorithm 1 is  $O\left(\frac{(2e)^k \cdot n \cdot |E| \log(1/\delta)}{\epsilon^2}\right)$  by Lemma 1, and by showing that the number of colorings used by the algorithm is  $O\left(\frac{e^k \log(1/\delta)}{\epsilon^2}\right)$ . This completes the proof. □

### 3.2 Counting $k$ -length Cycles with a chord



In this section assume that  $H$  is a simple cycle of length  $k$  with a chord. We present an algorithm to approximately compute the number of subgraphs of  $G$  which are isomorphic to  $H$ , and, for every vertex  $v$ , the number of subgraphs of  $G$  which are isomorphic to  $H$  and adjacent to  $v$ .

We first approximate number of colorful subgraphs of  $G$  which 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 Section 3.1. 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  $\ell$  the distance between the endpoint of the chord  $u, v$  on the cycle, and let  $A_{uv}^\ell = \{(S_1, S_2) | S_1, S_2 \subseteq [k], S_1 \setminus \{col(v), col(u)\} \cap S_2 \setminus \{col(v), col(u)\} = \phi, |S_1| = \ell + 1, |S_2| = k - \ell + 1\}$ .

Consider the following algorithm. The algorithm takes as input: a graph  $G = (V, E)$ , an approximation factor  $\epsilon$ , and an error probability  $\delta$ .

**Algorithm 3** (A  $(\epsilon, \delta)$ -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  $Z \cdot k^k / k!$ .

Our main theorem is the following:

**Theorem 2** 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 a  $(\epsilon, \delta)$ -approximation for the number of copies of  $H$  in  $G$ , with time complexity  $O\left(\frac{|E| \cdot n \cdot (2e)^k \cdot \log(1/\delta)}{\epsilon^2}\right)$ .

For proving Theorem 2 we first prove the following lemma:

**Lemma 2**  $X_i$  can be computed 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  $\ell$  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  $col(v)$  and  $col(u)$  are the only colors in the intersection of  $S_1$  and  $S_2$ . The number of colorful subgraphs of  $G$  which 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  $A_{uv}^\ell = \{(S_1, S_2) | S_1, S_2 \subseteq [k], S_1 \setminus \{col(v), col(u)\} \cap S_2 \setminus \{col(v), col(u)\} = \emptyset, |S_1| = \ell + 1, |S_2| = k - \ell + 1\}$ . The time complexity of computing  $X_i$ : By the proof of Lemma 1 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 for every  $(u, v) \in E$  and every  $2 \leq \ell \leq k - 2$   $|A_{uv}^\ell|$ . 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 2.** The correctness of the approximation returned by Algorithm 3 follow the technique of Alon *et al.* [2]. Lemma 2 implies the correctness of the computation of  $X_i$ . The time complexity of Algorithm 3 is  $O\left(\frac{|E| \cdot n \cdot (2\epsilon)^k \log(1/\delta)}{\epsilon^2}\right)$  by Lemma 2 and by showing that the number of colorings used by the algorithm is  $O\left(\frac{e^k \log(1/\delta)}{\epsilon^2}\right)$ . This completes the proof.  $\square$

We now approximate for every  $v \in V$  the number of colorful subgraphs of  $G$  which are isomorphic to  $H$  and are adjacent to  $v$ . Let  $t = \log(1/\delta)$ , let  $s = \frac{4 \cdot k^k}{\epsilon^2 k!}$ . 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  $A_{v'}^{z,b}(S) = \{(S_1, S_2) | S_1, S_2 \subseteq [k], |S_1| = z + 1, |S_2| = b - z + 1, S_1 \cup S_2 = S, S_1 \setminus \{col(u) | u \in V'\} \cap S_2 \setminus \{col(u) | u \in V'\} = \emptyset\}$ . Consider the following algorithm. The algorithm takes as input: a graph  $G = (V, E)$ , a vertex  $v$ , an approximation factor  $\epsilon$ , and an error probability  $\delta$ .

**Algorithm 4** (A  $(\epsilon, \delta)$ -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  $(u, w) \in E$ :

iv. For all  $S \subseteq [k]$  s.t  $|S| = \ell + 1$

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

v. Let

$$\begin{aligned} X_i^v &= X_i^v + \sum_{(u,v) \in E} \sum_{\ell=2}^{k-2} \sum_{(S_3, S_4) \in A_{uw}^{\ell, k}([k])} P_i(v, u, w, S_3) \cdot C_i(u, w, S_4) \\ &+ \sum_{(u,v) \in E} \sum_{\ell=2}^{k-2} \sum_{(S_3, S_4) \in A_{uw}^{k-\ell, k}([k])} P_i(v, u, w, S_3) \cdot C_i(u, w, S_4) \\ &+ \sum_{u \in N(v)} \sum_{\ell=2}^{k/2} \sum_{(S_3, S_4) \in A_{uv}^{\ell, k}([k])} C_i(v, u, S_3) \cdot C_i(v, u, S_4) \quad (1) \end{aligned}$$

(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 is the following:


**Theorem 3** 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 a  $(\epsilon, \delta)$ -approximation for the number of copies of  $H$  in  $G$  that are adjacent to  $v$ , with time complexity  $O\left(\frac{|E| \cdot n \cdot (3\epsilon)^k \log(1/\delta)}{\epsilon^2}\right)$ .


For proving Theorem 3 we first prove the following lemma:

**Lemma 3**  $X_i^v$  can be computed 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  $v$  is part of of length  $\ell$  between  $u$  and  $w$ , 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 multiply it by the number of colorful paths of length  $\ell - z$  between  $v$  and  $u$ , where

$1 \leq z \leq \ell - 1$ . (W.l.o.g assume that  $\ell \neq k - \ell$ ).  Thus for all  $S \subseteq [k]$

s.t  $|S| = \ell + 1$   $P_i(v, u, w, S) = \sum_{z=1}^{\ell-1} \sum_{(S_1, S_2) \in A_v^{z, \ell}(S)} C_i(v, w, S_1) \cdot C_i(v, u, S_2)$ .  
Therefore the total number of copies of  $H$  that are adjacent to  $v$  in the first case is the number of  $\ell$ -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 set of colors (except for the colors of  $u$  and  $w$ ):

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

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  $\ell$ -length colorful paths between  $u$  and  $v$  and multiply it by the number of  $(k - \ell)$ -length colorful paths between  $u$  and  $w$ , using disjoint set of colors besides the colors of  $u$  and  $v$ . This is done for all  $2 \leq \ell \leq k/2$ . Computing the running time: according to the proof of Lemma 1, 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  $C_i(v, w, S)$  is already computed) is

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

Therefore the time complexity of computing the first case is

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

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




## 4 Algorithms for Counting all four-size Motifs

Given a graph  $G = (V, E)$  and a vertex  $v$ , we describe how to approximately count for every vertex  $v$  the number of non-induced occurrences of each possible

motif  $H$  appearing in Pinter et al. [22] that are adjacent to  $v$ . In addition, for each motif  $H$  that appears in Pinter et al. [22] we present an algorithm for approximating the number of non-induced 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  $G$  which are isomorphic to  $H$ , and, for every vertex  $v$ , approximates the number of subgraphs of  $G$  which are isomorphic to  $H$  and adjacent to  $v$ .


We first approximate the later. There are three cases: one for which  $v$  is an endpoint of the path and adjacent to the triangle , second for which  $v$  is *not* an endpoint of the path and adjacent to the triangle , and third for which  $v$  is an endpoint of the path but *not* adjacent to the triangle . Let  $TR_G(v)$  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 received by removing all edges adjacent to  $v$ . Consider the following algorithm. The algorithm takes as input: a graph  $G = (V, E)$ , a vertex  $v$ , an approximation factor  $\epsilon$ , and an error probability  $\delta$ . Let  $TL_G(v)$  be the number of "tailed triangles" in  $G$  returned by the algorithm.

**Algorithm 5** (A  $(\epsilon, \delta)$ -approximation algorithm for counting simple "tailed triangles" adjacent to  $v$ )


1.  $TL_G(v) = 0$
2.  $TR_G(v) = \text{result of Algorithm 1 } (G, v, k = 3, \epsilon, \delta)$ .
3.  $TL_G(v) = TL_G(v) + TR_G(v) \cdot (|N(v)| - 2)$ .
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  $w \in \ell(u, v)$   $TL_G(v) = TL_G(v) + \deg w - 2 + \deg u - 2$ .
5. Compute  $G_v$  by going over the whole adjacency list and removing  $v$  any time it appears in the list.
6. For all  $u \in N(v)$   $TR_{G_v}(u) = \text{result of Algorithm 1 } (G_v, u, k = 3, \epsilon, \delta)$ .
7.  $TL_G(v) = TL_G(v) + \sum_{u \in N(v)} TR_{G_v}(u)$ .
8. Return  $TL_G(v)$ .

**Theorem 4** 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  $(\epsilon, \delta)$ -approximated, with time complexity  $O\left(\frac{n \cdot |E| \log(1/\delta)}{\epsilon^2}\right)$ .


**Proof:**

In the first case  we get that the number of subgraphs of  $G$  which are isomorphic to  $H$  and adjacent to  $v$  is

$$TR_G(v) \cdot (|N(v)| - 2).$$

In the second case  we get that the number of subgraphs of  $G$  which 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  $G$  which are isomorphic to  $H$  and adjacent to  $v$  is

$$\sum_{u \in N(v)} TR_{G_v}(u).$$

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

$$TR_G(v) \cdot (|N(v)| - 2) + \sum_{u \in N(v)} \sum_{w \in N(v) \cap N(u)} (\deg w - 2 + \deg u - 2) + \sum_{u \in N(v)} TR_{G_v}(u).$$

Let  $\hat{r}_v$  be the approximated value for the number of subgraphs of  $G$  which are isomorphic to  $H$  and are adjacent to  $v$  in the first and third cases. Let  $r_v$  be the exact number. In a similar manner to that of Theorem 1  $\hat{r}_v$  is an  $(\epsilon, \delta)$ -approximation to the number of subgraphs of  $G$  which are isomorphic to  $H$  and are 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  $G$  which are isomorphic to  $H$  and are adjacent to  $v$  contributed by this case. We need to show that  $\hat{r}_v + a_v$  is an  $(\epsilon, \delta)$ -approximation for the *total* number of subgraphs of  $G$  which are isomorphic to  $H$  and are adjacent to  $v$ :

$$\begin{aligned} & \Pr[\hat{r}_v + a_v \in [(1 - \epsilon)(r_v + a_v), (1 + \epsilon)(r_v + a_v)]] \\ & \geq \Pr[\hat{r}_v + a_v \in [(1 - \epsilon)r_v + a_v, (1 + \epsilon)r_v + a_v]] \geq 1 - 2\delta. \end{aligned} \quad (4)$$

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  $G$  which 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 1 the time complexity of computing the number of triangles that are adjacent to  $v$  for every  $v$  is  $O(\frac{n \cdot |E| \log(1/\delta)}{\epsilon^2})$ . 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

$$\begin{aligned} & O\left(\sum_{v \in V} \sum_{u \in N(v)} \deg u + \deg v\right) = O\left(\sum_{v \in V} \sum_{u \in N(v)} n\right) \\ & = O\left(n \cdot \sum_{v \in V} \deg v\right) = O(|E| \cdot n). \end{aligned} \quad (5)$$

Thus the total time complexity is

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

□


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


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

Assuming that for every  $v$   $\deg v$  is known, by Theorem 4 the time complexity is  $O(\frac{n \cdot |E| \log(1/\delta)}{\epsilon^2})$ . Therefore we immediately get the following theorem:

**Theorem 5** *Let  $G = (V, E)$  be an undirected graph, and let  $H$  be a triangle with a "tail" of length one. Then, the number of copies of  $G$  that are isomorphic to  $H$  can be  $(\epsilon, \delta)$ -approximated, with time complexity  $O(\frac{n \cdot |E| \log(1/\delta)}{\epsilon^2})$ .*

## 4.2 Counting 4-Cliques

In this section assume that  $H$  is a clique of size four.  We present an algorithm that *exactly* computes the number of subgraphs of  $G$  which are isomorphic to  $H$ , and, for every vertex  $v$ , the number of subgraphs of  $G$  which are isomorphic to  $H$  and adjacent to  $v$ .

We first compute, for every vertex  $v$ , the number of subgraphs of  $G$  which are isomorphic to  $H$  and adjacent to  $v$ . . We run the following algorithm: Let  $Cl(v)$  be the number of four-cliques in the graph that are adjacent to  $v$ . The algorithm takes as input: a graph  $G = (V, E)$ , a vertex  $v$ .

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

1.  $Cl(v) = 0$ .
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 if  $t \in \ell(u, v)$ . If  $t \in \ell(u, v)$  then  $Cl(v) := Cl(v) + 1$ .
3. Return  $Cl(v)/6$ .

**Theorem 6** 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  $O(|E| \cdot n \log n + |E|^2)$ .

**Proof:** The correctness of Algorithm 6 is trivial. Using similar computation to that in the proof of Theorem 4 we get that the time complexity of Algorithm 6 is

$$\begin{aligned}
& O\left(\sum_{v \in V} \sum_{u \in N(v)} \deg u + \deg v\right) \\
& + O\left(\sum_{v \in V} \sum_{u \in N(v)} |N(v) \cap N(u)| \log(|N(v) \cap N(u)|) + \sum_{w \in N(v) \cap N(u)} \deg w\right) \\
& = O(|E| \cdot n \log n) + O\left(\sum_{v \in V} \sum_{u \in N(v)} \sum_{w \in N(v) \cap N(u)} \deg w\right) \\
& = O(|E| \cdot n \log n) + O\left(\sum_{v \in V} \sum_{u \in N(v)} \sum_{w \in V} \deg w\right) \\
& = O(|E| \cdot n \log n) + O(|E|^2). \tag{6}
\end{aligned}$$

□



We now count the number of subgraphs of  $G$  which 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:

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

**Theorem 7** 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  $O(|E| \cdot n \log n + |E|^2)$ .

### 4.3 Counting Small Trees

In this section assume that  $H$  is a tree of size four that is consisted of a vertex and three of its neighbors. We present an algorithm that *exactly* computes, for every vertex  $v$ , the number of subgraphs of  $G$  which 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  $G$  which are isomorphic to  $H$  and are adjacent to  $v$ . Thus the total number number of subgraphs of  $G$  which are isomorphic to  $H$  and are 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 8** *Let  $G = (V, E)$  be an undirected graph, and let  $H$  be a tree of size four that is consisted of a vertex and three of its neighbors. Then, for every vertex  $v$ , the number of subgraphs of  $G$  which are isomorphic to  $H$  and are adjacent to  $v$  can be found with time complexity  $O(|E| + n)$ .*

Note that the total number of subgraphs of  $G$  which 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 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 non-induced occurrences of the motif the vertex is part of, for  $k$ -length cycles and  $k$ -length cycles with a chord, where  $k = O(\log n)$ . We also designed algorithms with time complexity  $O\left(\frac{n \cdot |E| \cdot \log \frac{1}{\delta}}{\epsilon^2} + |E|^2 + |E| \cdot n \log n\right)$  that, for the first time, approximate for every vertex the number of non-induced occurrences of the motif the vertex is part of, for all motifs of size of at most four. In addition, we showed algorithms that approximate the total number of non-induced occurrences of these network motifs, when no efficient algorithm exists. Approximating the number of non-induced occurrences of the motif a vertex is part of, for other motifs of size  $O(\log n)$  is left for future work.

**Acknowledgment:** we thank Dana Ron for many hours of fruitful discussions, and the anonymous reviewers for their help in improving this manuscript. This work was support by a center of excellence on “Network Topology: structure and dynamics” grant (No. 1685/07) by the Israeli Science Foundation.

## References

1. R. Albert and A.-L. Barabási. Topology of evolving networks: Local events and universality. *85(24):5234–5237*, 2000.
2. N. Alon, P. Dao, I. Hajirasouliha, F. Hormozdiari, and S. C. Sahinalp. Biomolecular network motif counting and discovery by color coding. *Bioinformatics*, 1:1–9, 2008.
3. N. Alon and S. Gutner. Balanced families of perfect hash functions and their applications. In *Proceedings of the 34th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 435–446, 2007.
4. N. Alon, R. Yuster, and U. Zwick. Color coding. *Journal of the ACM*, 42:844–856, 1995.
5. N. Alon, R. Yuster, and U. Zwick. Finding and counting given length cycles. *Algorithmica*, 17:209–223, 1997.
6. V. Arvind and V. Raman. Approximation algorithms for some parameterized counting problems. In *Proceedings of the 13th International Symposium on Algorithms and Computation (ISAAC)*, pages 453–464, 2002.
7. S. Bar, M. Gonen, and A. Wool. An incremental super-linear preferential Internet topology model. In *Proc. 5th , LNCS 3015*, pages 53–62, Antibes Juan-les-Pins, France, 2004.
8. S. Bar, M. Gonen, and A. Wool. A geographic directed preferential Internet topology model. In *Proc. 13th*, pages 325–328, Atlanta, GA, 2005.
9. P. Barford, A. Bestavros, J. Byers, and M. Crovella. On the marginal utility of network topology measurements. In *Proc.*, pages 5–17, San Francisco, California, USA, 2001.
10. L. Becchetti, P. Boldi, C. Castillo, and A. Gionis. Efficient semi-streaming algorithms for local triangle counting in massive graphs. In *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*, pages 16–24, 2008.
11. G. Bianconi and A. L. Barabási. Competition and multiscaling in evolving networks. *Europhysics Letters*, 54(4):436–442, 2001.
12. G. Bianconi and A. Capocci. Number of loops of size  $h$  in growing scale-free networks. *Physical Review Letters*, 90:078701, 2003.
13. G. Bianconi and M. Marsili. Loops of any size and hamilton cycles in random scale-free networks. *Journal of Statistical Mechanics*, page P06005, 2005.
14. G. Bianconi and M. Marsili. Number of cliques in random scale-free network ensembles. *Physica D*, 224:1–6, 2006.
15. R.X. Brunet and I.M. Sokolov. Evolving networks with disadvantaged long-range connections. *Physical Review E*, 66:026118, 2002.
16. T. Bu and D. Towsley. On distinguishing between Internet power-law generators. In *Proc. IEEE INFOCOM'02*, pages 638–647, New-York, NY, USA, 2002.
17. Q. Chen, H. Chang, R. Govindan, S. Jamin, S. Shenker, and W. Willinger. The origin of power laws in Internet topologies revisited. In *Proc. IEEE INFOCOM'02*, pages 608–617, New-York, NY, USA, 2002.
18. B. Dost, T. Shlomi, N. Gupta, E. Ruppín, V. Bafna, and R. Sharan. QNet: A tool for querying protein interaction networks. In *Proceedings of the 11th Annual International Conference Research in Computational Molecular Biology (RECOMB)*, pages 1–15, 2007.
19. 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):598–620, 1995.

20. C. Faloutsos, M. Faloutsos, and P. Faloutsos. On power-law relationships of the Internet topology. In *Proc. of ACM SIGCOMM'99*, pages 251–260, Cambridge, Massachusetts, USA, 1999.
21. Dima Feldman and Yuval Shavitt. Automatic large scale generation of internet pop level maps. In *IEEE GLOBECOM*, 2008.
22. M. Gordon, E. A. Livneh, R. Y. Pinter, and E. Rubin. Elucidating protein function using graphlet degree vectors in protein-protein interactions networks. under review.
23. R. Govindan and H. Tangmunarunki. Heuristics for Internet map discovery. In *Proc. IEEE INFOCOM'00*, pages 1371–1380, Tel-Aviv, Israel, 2000.
24. J. Grochow and M. Kellis. Network motif discovery using subgraph enumeration and symmetry-breaking. In *Proceedings of the 11th Annual International Conference Research in Computational Molecular Biology (RECOMB)*, pages 92–06, 2007.
25. 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):239–249, 2008.
26. 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):e118, 2007.
27. R. Itzhack, Y. Mogilevski, and Y. Louzoun. An optimal algorithm for counting network motifs. *Physica A*, 381:482–490, 2007.
28. 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 (FOCS)*, pages 56–64, 1983.
29. N. Kashtan, S. Itzkovitz, R. Milo, and U. Alon. Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs. *Bioinformatics*, 20(11):1746–1758, 2004.
30. V. Van Kerrebroeck and E. Marinari. Ranking by loops: a new approach to categorization. *Physical Review Letters*, 101:098701, 2008.
31. A. Lakhina, J. W. Byers, M. Crovella, and P. Xie. Sampling biases in IP topology measurements. In *Proc. IEEE INFOCOM'03*, pages 332–341, New-York, NY, USA, 2003.
32. X. Li and G. Chen. A local-world evolving network model. *Physica A*, 328:274–286, 2003.
33. E. Marinari, G. Semerjian, and V. Van Kerrebroeck. Finding long cycles in graphs. *Physical Review E*, 75(6):066708, June 2007.
34. R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. Network motifs: Simple building blocks of complex networks. *Science*, 298:824–827, 2002.
35. N. Przulj. Biological network comparison using graphlet degree distribution. *Bioinformatics*, 23(2):e177–e183, 2007.
36. N. Przulj, D. G. Corneil, and I. Jurisica. Modeling interactome: Scale-free or geometric? *Bioinformatics*, 20(18):3508–3515, 2004.
37. H. Reittu and I. Norros. On the power law random graph model of the Internet. *Performance Evaluation*, 55, 2004.
38. G. Sagie and A. Wool. A clustering approach for exploring the Internet structure. In *Proc. 23rd*, pages 149–152, Herzlia, Israel, 2004.
39. J. Scott, T. Ideker, R. Karp, and R. Sharan. Efficient algorithms for detecting signaling pathways in protein interaction networks. In *Proceedings of the 9th An-*

- nual International Conference Research in Computational Molecular Biology (RECOMB)*, pages 1–13, 2005.
40. Yuval Shavitt and Eran Shir. DIMES: Let the internet measure itself. *ACM SIGCOMM Computer Communication Review*, 35:71–74, October 2005.
  41. T. Shlomi, D. Segal, and E. Ruppin. QPath: a method for querying pathways in a protein-protein interaction network. *Bioinformatics*, 7:199, 2006.
  42. G. Siganos, S.L. Tauro, and M. Faloutsos. Jellyfish: A conceptual model for the as Internet topology. *Journal of Communications and Networks*, 8:339–350, 2006.
  43. L. Subramanian, S. Agarwal, J. Rexford, and R. H. Katz. Characterizing the Internet hierarchy from multiple vantage points. In *Proc. IEEE INFOCOM'02*, pages 618–627, New-York, NY, USA, 2002.
  44. H. Tangmunarunkit, R. Govindan, S. Jamin, S. Shenker, and W. Willinger. Network topology generators: Degree based vs. structural. In *Proc.*, pages 147–159, New York, USA, 2002.
  45. L. Tauro, C. Palmer, G. Siganos, and M. Faloutsos. A simple conceptual model for Internet topology. In *Proc. IEEE Global Internet*, pages 1667–1671, San Antonio, TX, 2001.
  46. S. Wernicke. Efficient detection of network motifs. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 3(4):347–359, October 2006.
  47. 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*, 99:2573–2580, 2002.
  48. 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.