BIOINFORMATICS

Vol. 24 ISMB 2008, pages i241–i249
doi:10.1093/bioinformatics/btn163

Biomolecular network motif counting and discovery by color coding

Noga Alon[1], Phuong Dao[2], Iman Hajirasouliha[2], Fereydoun Hormozdiari[2] and S. Cenk Sahinalp[2,*]

[1]Schools of Mathematical Sciences and Computer Science, Tel Aviv University, Ramat Aviv, Israel and
[2]School of Computing Science, Simon Fraser University, Burnaby, BC, Canada

ABSTRACT

Protein–protein interaction (PPI) networks of many organisms share *global topological features* such as degree distribution, k-hop reachability, betweenness and closeness. Yet, some of these networks can differ significantly from the others in terms of *local structures*: e.g. the number of specific network motifs can vary significantly among PPI networks.

Counting the number of network motifs provides a major challenge to compare biomolecular networks. Recently developed algorithms have been able to count the number of *induced* occurrences of subgraphs with $k \leq 7$ vertices. Yet no practical algorithm exists for counting *non-induced* occurrences, or counting subgraphs with $k \geq 8$ vertices. Counting non-induced occurrences of network motifs is not only challenging but also quite desirable as available PPI networks include several false interactions and miss many others.

In this article, we show how to apply the 'color coding' technique for counting non-induced occurrences of subgraph topologies in the form of trees and bounded treewidth subgraphs. Our algorithm can count all occurrences of motif G' with k vertices in a network G with n vertices in time polynomial with n, provided $k = O(\log n)$. We use our algorithm to obtain 'treelet' distributions for $k \leq 10$ of available PPI networks of unicellular organisms (*Saccharomyces cerevisiae Escherichia coli* and *Helicobacter Pyloris*), which are all quite similar, and a multicellular organism (*Caenorhabditis elegans*) which is significantly different. Furthermore, the treelet distribution of the unicellular organisms are similar to that obtained by the 'duplication model' but are quite different from that of the 'preferential attachment model'. The treelet distribution is robust w.r.t. sparsification with bait/edge coverage of 70% but differences can be observed when bait/edge coverage drops to 50%.

Contact: cenk@cs.sfu.ca

1 INTRODUCTION

Recent research has revealed that many biomolecular networks share global topological features. Similarities between protein–protein interaction (PPI) networks of several organisms have been observed with respect to their degree distribution, k-hop reachability, betweenness and closeness (Bebek *et al.*, 2006; Bollobás *et al.*, 2001; Hormozdiari *et al.*, 2007; Przulj *et al.*, 2004).

Topological similarities have also been observed between PPI networks and networks generated by random processes. For example, the degree distribution of the 'preferential attachment model' is similar to that of the Yeast (*S.cerevisiae*) PPI network (Eisenberg and Levanon, 2003). More interestingly, the 'duplication model' generates networks that are very similar to the PPI networks of a number of organisms (including that of the Yeast) not only in terms of degree distribution but also k-hop reachability (for $k \leq 6$), betweenness and closeness (Hormozdiari *et al.*, 2007). Because direct measures for comparing two networks, such as the minimum number of edges and vertices to be deleted to make two networks isomorphic are NP-hard to compute, such topological features have been used to 'measure' how similar any given pair of networks could be.

Two networks which have similar global features can have significant differences in terms of local structures they include: e.g. one of them may include a specific subgraph many more times than the other. Thus, it is important to be able to count the 'number of occurrences' of specific subgraphs in networks as means of detecting whether two networks are similar or not.

A subgraph that occurs much more frequently in a biomolecular network G than one in a 'random' network or a 'typical' network R whose global properties are similar to those of G is called a network motif of G (Milo *et al.*, 2002). Similarly, a subgraph that occurs much less frequently in G in comparison to R is called an anti-motif of G.

The use of subgraph distribution with up to k vertices to compare PPI networks with artificial networks has been the source of a recent debate. It was argued that the distribution of subgraphs of up to $k = 5$ vertices in the Yeast PPI network is quite different from that of the preferential attachment model (Przulj *et al.*, 2004). Based on this observation, it was argued that the Yeast PPI network is not a 'scale-free' network and the presumed similarity of the Yeast PPI network and the 'scale-free' networks in terms of degree distribution is a consequence of sampling errors (Han *et al.*, 2005). Finally, in Hormozdiari *et al.* (2007) it was demonstrated that the subgraph distribution of the preferential attachment model and that of the duplication model for $k \leq 6$ can be substantially different and the seed network of the duplication method could be chosen in a way that its subgraph distribution can be made 'very similar' to that of the available PPI networks including that of the Yeast.

Although it is possible to make the general distribution of subgraphs in an artificial model (more specifically the duplication model) very similar to that of a specific PPI network, there are a number of subgraphs, for example, in the Yeast PPI network, which occur much more frequently than that in the associated artificial model. These motifs were suggested to be recurring circuit elements that carry out key information processing tasks (Milo *et al.*, 2002), and thus are of considerable interest. As a result, novel computational tools have been developed for counting subgraphs in a network

*To whom correspondence should be addressed.

(Hormozdiari *et al.*, 2007; Przulj *et al.*, 2004) and discovering network motifs (Grochow and Kellis, 2007).

Counting the number of all possible 'induced' subgraphs in a PPI network is a very challenging task. Przulj *et al.* (2004) describes how to count all induced subgraphs with up to $k=5$ vertices in a PPI network. Faster techniques that count induced subgraphs of size up to $k=6$ (Hormozdiari *et al.*, 2007) and $k=7$ (Grochow and Kellis, 2007) were developed very recently. The running time of these techniques all increase exponentially with k. Thus novel algorithmic tools are now needed for counting subgraphs of size $k \geq 8$.

An alternative approach to 'estimate' the number of specific induced subgraphs with k vertices is through the sampling strategy suggested by Kashtan *et al.* (2004). This sampling strategy is based on a random walk approach, which, in k iterations, picks k vertices of the input network and includes all the edges between the vertices picked. Although this strategy has not been proven to work for all subgraphs and all input networks, it has been experimentally shown to be accurate for specific subgraphs even when a small number of samples are used (Kashtan *et al.*, 2004).

Note that an induced subgraph (more accurately a vertex induced subgraph) of a network G is a subset of the vertices of G together with any edges whose endpoints are both in this subset; i.e. G' is an induced subgraph of G if and only if for each pair of vertices v' and w' in G' and their corresponding vertices v and w in G, either there are edges between both v', w' pair and v, w pair or there are no edges between any of the pairs. For example, let G be a fully connected network of size n. Then a cycle that goes through every vertex in G is not an induced subgraph of G; it is called a 'non-induced' subgraph of G.

All the above techniques consider only induced subgraphs of a given network; there are many more non-induced subgraphs isomorphic to a given topology and thus it is more difficult to count non-induced subgraphs of a network. As a result, there are only a limited number of earlier studies on biomolecular networks that consider non-induced subgraphs (Dost *et al.*, 2007; Scott *et al.*, 2006). The motivation for considering non-induced subgraphs are clear: available PPI networks are far from complete and error free; the interactions between proteins reported by these networks include both false positives and false negatives. Thus, an occurrence of a specific network motif in one network may include additional edges in its occurrence in another network and vice versa.

The specific problem addressed by earlier studies on non-induced subgraphs (Dost *et al.*, 2007; Scott *et al.*, 2006) is not the subgraph counting problem. Rather these papers focus on the 'subgraph detection' problem, which aims to respond to queries of the form, does an input network G have a non-induced subgraph G'—where G' is a user specified query subgraph. Subgraph detection problem is somewhat easier than the subgraph counting problem. Dost *et al.* (2007), for example, show how to solve the subgraph detection problem for subgraphs of size $k=O(\log n)$—much larger than what can be tackled by Grochow and Kellis (2007); Hormozdiari *et al.* (2007); Przulj *et al.* (2004) for subgraph counting—provided that the query subgraph G' is either a simple path, a tree or a bounded treewidth subgraph. The main tool employed here that makes subgraph detection problem tractable for such subgraphs is the 'color coding' technique (Alon *et al.*, 1995).

Color coding is a combinatorial approach that was introduced to detect simple paths, trees and bounded treewidth subgraphs in unlabeled graphs (Alon *et al.*, 1995). It was later applied to subgraph detection in biomolecular networks by Shlomi *et al.* (2006) and Dost *et al.* (2007). Color coding is based on assigning random colors to the vertices of an input graph. For subgraph detection purposes, it considers only those subgraphs where each vertex has a unique color as a potential answer to a query subgraph. Such 'colorful' subgraphs which are isomorphic to the query subgraph can then be detected through efficient use of dynamic programming, in time polynomial with n, the size of the input network. If the above procedure is repeated sufficiently many times (polynomial with n, provided that the subgraph we are looking for is of size $k=O(\log n)$), it is guaranteed that a specific occurrence of the query subgraph will be detected with high probability.

Arvind and Raman (2002) use the color coding approach to count the number of subgraphs in a given network G, which are isomorphic to a *bounded treewidth graph* H. They give a randomized approximate counting algorithm with running time $k^{O(k)} \cdot n^{b+O(1)}$ where n and k are the number of vertices in G and H, respectively, and b is the treewidth of H. The framework which they use is based on (Karp and Luby, 1983) for approximate counting via sampling. Provided that $k=O(\log n)$, the running time of this algorithm is *super-polynomial* with n, and thus is not practical.

(Alon and Gutner, 2007) combines the color coding technique with a construction of what is called *Balanced Families of Perfect Hash Functions* to obtain a *deterministic* algorithm to count the number of *simple paths or cycles* of size k in an input network G. This algorithm has a running time of $2^{O(k \log\log k)} n^{O(1)}$, still *super-polynomial* in n when $k=O(\log n)$.

1.1 Our contributions

Given a network with n vertices, we show how to apply the color coding technique to *count* non-induced trees and bounded treewidth subgraphs with k vertices. We present a randomized approximation algorithm with running time $2^{O(k)} \cdot n^{O(1)}$, which is polynomial in n for $k=O(\log n)$ and thus is faster than available alternatives (Alon and Gutner, 2007; Arvind and Raman, 2002). Our algorithm is quite efficient in practice; we were able to go beyond what the algorithms presented in Grochow and Kellis (2007); Hormozdiari *et al.* (2007); Przulj *et al.* (2004) achieve, and count, for the first time, *all* possible tree topologies of 8, 9 and 10 vertices in PPI networks of various organisms such as *S.Cerevisiae* (Yeast), *E.coli*, *H.pylori* and *C.elegans* (Worm) PPI networks available via the DIP database (Xenarios *et al.*, 2002).[1] We also compare these networks with artificial networks generated by the Preferential Attachment Model (Barabási and Albert, 1999; Bollobás *et al.*, 2001; Chung *et al.*, 2001) and the Duplication Model (Bebek *et al.*, 2006; Chung *et al.*, 2003; Vázquez *et al.*, 2003).

The distribution of bounded treewidth subgraphs, and in particular trees of up to 10 vertices provides us powerful means to compare biomolecular networks. One of the important features of what we call the 'normalized treelet distribution', the distribution of the number of occurrences of non-induced trees in a PPI network, normalized by the total number of such treelets, is that it is quite robust. On the well-known Yeast PPI network (Xenarios *et al.*, 2002), even after random sparsification with bait coverage of 70% and edge coverage of 70% (as suggested by Han *et al.*, 2005),

[1]The DIP release date for the full Yeast, *E.coli* and *H.pylori* PPI networks is July 7, 2007. For the Core yeast network, we use the network which was released on July 29, 2007

the normalized tree distribution does not change much. However, no means of graph comparison should be too robust w.r.t. sparsification—otherwise it cannot illustrate the differences between a pair of networks that are very similar, and those which are not. The normalized treelet distribution is indeed not robust to an extreme; after sparsifying the Yeast PPI network with 50% bait and 50% edge coverage, differences become significant.

It is interesting to note that the normalized treelet distributions of the three unicellular organisms we compared, Yeast, *E.coli* and *H.pylori* were all fairly similar; however, the distribution of the more complex *C.elegans* was quite different. Furthermore, the normalized treelet distribution of the artificial graphs generated by the Duplication Model is quite close to that of the three unicellular organisms we tested but the distribution of the preferential attachment model has some noticeable differences.

2 THE SUBGRAPH COUNTING ALGORITHM

In this section, we describe how to apply the color coding technique to approximately count the number of non-induced occurrences of each possible tree topology T with $O(\log n)$ vertices in a network G with n vertices. As per Arvind and Raman (2002), this method can be generalized to count all non-induced occurrences of each bounded treewidth graph G' in G as well, provided that the treewidth is constant.

Given a network G with n vertices and a tree T with k vertices, we consider the problem of counting the number of non-induced subtrees of G that are isomorphic to T. Note that we use the standard definition of a tree, i.e. for us, a tree is an unlabeled, connected graph with no cycles. It is unrooted and its vertices are unordered.[2] A tree T is said to be isomorphic to a subtree T' in a network G if there is a bijection between the vertices of T and the vertices of T' such that for every edge between two vertices a and b of T there is an edge between the vertices a' and b' in T' that correspond to a and b, respectively. Such a tree T' is considered to be a non-induced occurrence of T in G.

Note that we allow overlaps between the trees we count, i.e. two occurrences of T, namely T' and T'' may share vertices; in fact the vertex sets of T' and T'' may be identical. We consider T' and T'' distinct occurrences of T provided that the edge sets of T' and T'' are not identical.

Our algorithm counts the number of non-induced occurrences of a tree T with $k = O(\log n)$ vertices in a network G with n vertices as follows.

1. **Color coding.** Color each vertex of input graph G independently and uniformly at random with one of the k colors.
2. **Counting.** Apply a dynamic programming routine (explained later) to count the number of non-induced occurrences of T in which each vertex has a unique color.
3. Repeat the above two steps $O(e^k)$ times and add up the number of occurrences of T to get an estimate on the number of its occurrences in G.

In what follows, we give the details of the above steps and explain how and why they work.

2.1 Color coding step

We note that the color coding step not only works for trees but also bounded treewidth graphs with constant treewidth—without any modifications. Let r be the total number of copies of T in G. We assign a color to each vertex of G from the color set $[k] = \{1, \ldots, k\}$. The colors are assigned to each vertex independently and uniformly at random. It is easy to see that for a particular non-induced occurrence of T in G the probability that all its vertices are assigned unique colors is $p = k!/k^k$, thus the expected number of colorful copies in G is rp.

Let $\mathcal{F}$ denote the family of all copies of T in G. For each such copy $F \in \mathcal{F}$, let x_F denote the indicator random variable whose value is 1 if and only if the copy is colorful in our random k-coloring of $V(G)$, the vertices of G. Let $X = \sum_{F \in \mathcal{F}} x_F$ be the random variable counting the total number of colorful copies of T. By linearity of expectation, the expected value of X is $E(X) = rp$.

It is possible to estimate the variance of X as follows. Note, first, that for every two distinct copies $F, F' \in \mathcal{F}$, the probability that both F and F' are colorful is at most p (and in fact strictly smaller unless both copies have exactly the same set of vertices), implying that the covariance $\mathrm{Cov}(x_F, x_{F'})$ satisfies

$$\mathrm{Cov}(x_F, x_{F'}) = E(x_F x_{F'}) - E(x_F)E(x_{F'}) \leq p.$$

Therefore, the variance of X satisfies

$$\mathrm{Var}(X) = \sum_{F \in \mathcal{F}} \mathrm{Var}(x_F) + \sum_{F \neq F' \in \mathcal{F}} \mathrm{Cov}(x_F, x_{F'})$$

$$\leq rp + r(r-1)p = r^2 p.$$

It follows that if Y is the average of s independent copies of X (obtained by s independent random colorings), then

$$E(Y) = E(X) = rp$$

and

$$\mathrm{Var}(Y) = \mathrm{Var}(X)/s \leq r^2 p/s.$$

Therefore, by Chebyshev's Inequality, the probability that Y is smaller than (or bigger than) its expectation by at least ϵrp is at most

$$\frac{r^2 p}{\epsilon^2 r^2 p^2 s} = \frac{1}{\epsilon^2 ps}.$$

In particular, if $s = 4/\epsilon^2 p$ this probability is at most $1/4$.

In case we wish to decrease the error probability, we can compute Y t times independently and let Z be the median. The probability that the median is less than $(1-\epsilon)rp$ is the probability that at least half of the copies of Y computed will be less than this quantity, which is at most

$$\binom{t}{t/2} 4^{-t} \leq 2^{-t}.$$

[2] Thus, for example, consider a tree T with a root vertex a with two children b and c, with b having a single child d. For our purposes T is isomorphic to another tree where b is the root with two children d and a, and a with a single child c. In fact, both of these trees are isomorphic to a simple path involving four vertices.

A similar estimate holds for the probability that Z is bigger than $(1+\epsilon)rp$. Therefore, if $t=\log(1/\delta)$ then with probability $1-2\delta$ the value of Z will lie in $[(1-\epsilon)rp,(1+\epsilon)rp]$. Note that the total number of colorings in the process is

$$O\left(\frac{\log(1/\delta)}{\epsilon^2 p}\right)=O\left(\frac{e^k\log(1/\delta)}{\epsilon^2}\right).$$

Our estimate for r is, of course, $Z/p=Zk^k/k!$.

2.2 Counting step

Given a random coloring of the input vertices with k colors, we present a dynamic programming algorithm to compute the number of colorful subgraphs of G which are isomorphic to the query tree T.

To give a flavor of our algorithm, we first present it for the case in which the query graph is a single path of length k. For each vertex v and each subset S of the color set $\{1,\ldots,k\}$, we aim to record the number of colorful paths for which one of the endpoints is v. Let $C(v,S)$ be the number of such paths, and col(v) be the color of vertex v. Given a color ℓ, for all $v\in V(G)$:

$$C(v,\{\ell\})=\begin{cases}1 & \text{if col}(v)=\ell\\ 0 & \text{otherwise.}\end{cases}$$

For each vertex v and color set S where $|S|>1$, we have

$$C(v,S)=\sum_{u;(u,v)\in E(G)} C\big(u,S-col(v)\big).$$

Note that the number of single colorful paths of length k would be

$$\frac{1}{2}\sum_v C\big(v,\{1,\ldots,k\}\big).$$

As mentioned earlier, we will only describe the counting step for the case T is a tree, however the algorithm we present can be generalized to bounded treewidth graphs with constant treewidth without much difficulty.

As a first step we pick an arbitrary vertex ρ of T and set it as the *root*. We will denote this rooted tree by $\tau(\rho)$. Then we count the number of colorful occurrences of $\tau(\rho)$ in the given graph G as follows.

For each vertex v of the graph G, we compute $c(v,\tau(\rho),[k])$, the number of $[k]$-colorful rooted subtrees with root v, which are isomorphic to $\tau(\rho)$.

The actual number of $[k]$-colorful occurrences of T in G is

$$\frac{1}{q}\sum_v c(v,\tau(\rho),[k])$$

where q is equal to the number of vertices u in T, for which the rooted tree $\tau(u)$ is isomorphic to $\tau(\rho)$.

In order to compute $c(v,\tau(\rho),[k])$ for every vertex v in the graph G, we use the following dynamic programming routine.

Let $\tau'(\rho')$ be a subtree of the tree $\tau(\rho)$ with root ρ', we denote the size of $\tau'(\rho')$ by $\nu(\tau'(\rho'))$. For any vertex x in G, and a subset S of the color set $[k]$ with $|S|=\nu(\tau'(\rho'))$, let $c(x,\tau'(\rho'),S)$ be the number of S-colorful subgraphs with root x and color set S, which are isomorphic to $\tau'(\rho')$. We compute $c(x,\tau'(\rho'),S)$ inductively as follows.

The base case where $\nu(\tau'(\rho'))=1$ is obvious: For any single color set $S=\{a\}$, $c(x,\tau'(\rho'),S)$ is equal to 1 if x has color a, and otherwise is equal to 0.

For the case where $\nu(\tau'(\rho'))\geq 2$, let ρ'' be a vertex connected to ρ' in $\tau'(\rho')$. Removing the edge (ρ',ρ'') partitions $\tau'(\rho')$ into two smaller subtrees, say $\tau'_1(\rho')$ with root ρ', and $\tau'_2(\rho'')$ with root ρ''.

Now for every vertex u connected to x in G, and all set of colors S_1 and $S_2\subset[k]$ with $|S_1|=\nu(\tau'_1(\rho'))$, $|S_2|=\nu(\tau'_2(\rho''))$ and $S_1\cap S_2=\emptyset$ we recursively find $c(x,\tau_1(\rho'),S_1)$ and $c(u,\tau_2(\rho''),S_2)$. The next step is to compute $c(x,\tau'(\rho'),S)$, by using the values of $c(x,\tau_1(\rho'),S_1)$ and $c(u,\tau_2(\rho''),S_2)$ for every u connected to x, and all feasible set of colors S_1 and S_2. This is easily achieved by the fact that

$$c(x,\tau'(\rho'),S)=$$
$$\frac{1}{d}\sum_{\forall S_1,S_2:|S_1\cap S_2|=\emptyset} c(x,\tau_1(\rho'),S_1)\cdot c(u,\tau_2(\rho''),S_2).$$

Here, d is the *over counting factor* and is equal to one plus the number of those siblings of ρ'' (i.e. vertices connected to ρ') which are roots of subtrees isomorphic to $\tau'(\rho'')$.

Note that the total running time of our algorithm would be polynomial in n. We need to repeat the experiment $O(e^k\log(1/\delta)/\epsilon^2$ times, and each counting step takes $O(2^k\cdot|E|)$ where $|E|$ is the number of edges in the input network. Thus, the asymptotic running time of our algorithm is

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

3 EXPERIMENTAL RESULTS

We tested our algorithm to count non-induced occurrences of subgraphs with $k=8,9,10$ vertices. Due to limits of computational resources, we have not been able to go beyond $k=10$. Table 1 shows the number of unlabeled tree topologies for different values of k together with the total running time of our algorithm for counting the non-induced occurrences of these trees in the largest connected component of the Yeast PPI network. We note that our algorithm is quite fast; for $k=10$, it takes 12 h to count all tree topologies on a Sun Fire X4600 Server with 64GB RAM, when executed in parallel on 8 dual AMD Opteron CPUs with 2.6 Ghz speed.[3] Furthermore, our algorithm is highly accurate in practice; as can be seen in Figure 1, our algorithm's estimates on the number of occurrences of each subgraph topology of $k=8$ is very close to their actual number of occurrences in the Core Yeast PPI network.

The list of tree topologies for various values of k can be obtained from the Combinatorial Object Server Generation website

Table 1. Number of unlabeled tree topologies, and the running time of our algorithm to count them in the Yeast PPI network

No. of vertices (k)	No. of unlabeled trees	Running time (mins)
7	11	2
8	23	14
9	47	100
10	106	700

[3]We do not provide a direct comparison of this method with alternative schemes such as Grochow and Kellis (2007) w.r.t. running time as we focus on counting non-induced occurrences of motifs whereas all alternative schemes focus on induced occurrences.

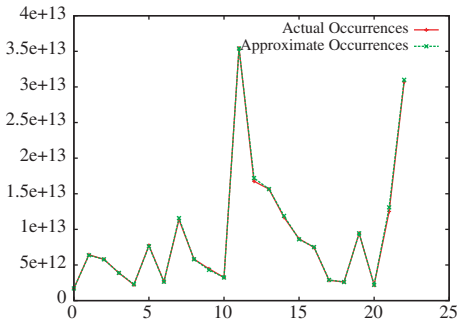

Fig. 1. Comparison between the output of our algorithm and the actual occurrences for subtrees of size $k=8$.

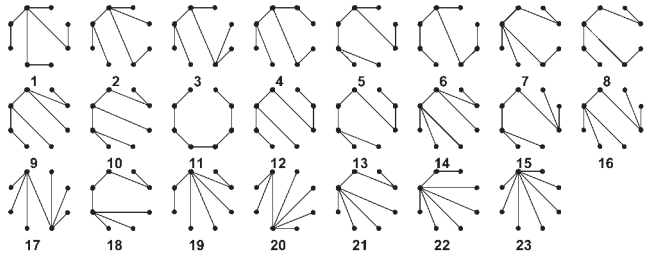

Fig. 2. List of treelets for $k=8$.

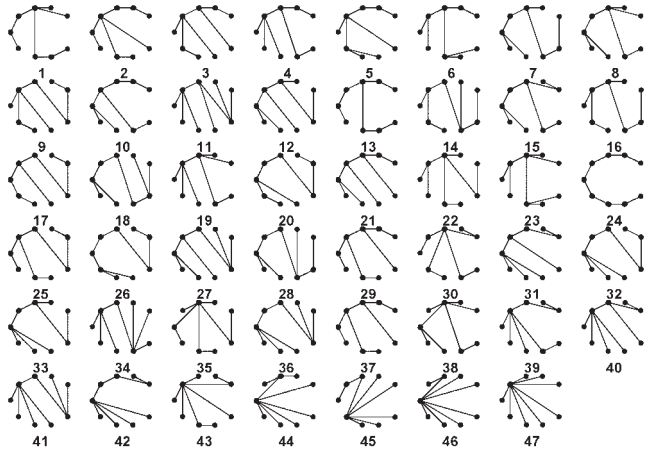

Fig. 3. List of treelets with $k=9$.

(http://theory.cs.uvic.ca/cos.html). Figures 2–4 depict all tree topologies for $k=8,9,10$ respectively.

We tested our algorithm on the protein–protein interaction networks of four species: *S.cerevisiae* (Yeast), *E.coli*, *H.pylori* and *C.elegans* (Worm). Since the PPI networks of these species are far from complete, we focus on the largest connected component of each network. For each PPI network and for all trees of $k=8,9,10$ vertices, we counted the number of non-induced occurrences of each tree topology. The distribution of the number of such subtree topologies (which will be called 'treelets') for varying values of k provide means of comparing PPI networks.

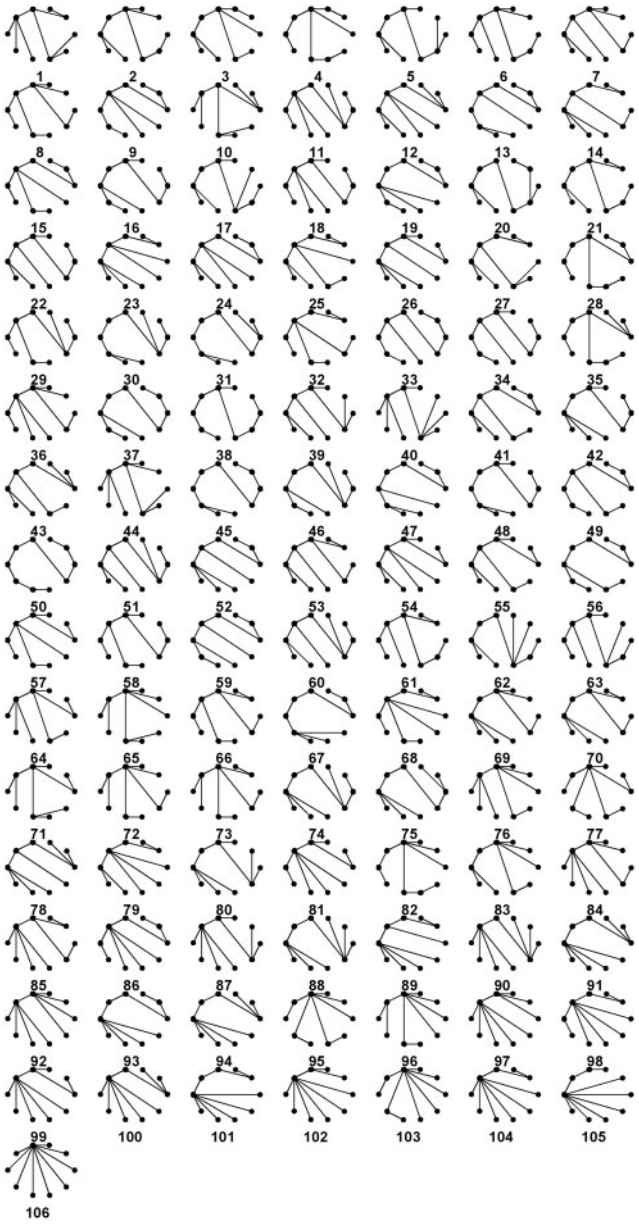

Fig. 4. List of treelets with $k=10$.

Note that the number of vertices, their average degree, etc. vary significantly from one PPI network to the other. Table 2 shows the number of vertices and edges of the PPI networks we used in our study. Thus, it should be expected that the number of non-induced occurrences of treelets should differ considerably among the networks.

As a result, we *normalize* the treelet distributions of each individual network, for each value of k as follows. For each treelet T of k vertices, consider the *fraction* of the number of occurrences of T in a network G among *total number of occurrences of all possible treelets* of size k in G. The normalized treelet distribution refers to this fractional count of treelets in a given PPI network. We note that as specific fractions of treelets vary by several orders of magnitude, our normalized treelet distributions are all given in logarithmic scale.

Table 2. Number of vertices, edges and average degree in the PPI networks we studied

Network	No. of vertices	No. of edges	Average degree
S.cerevisiae	2345	5609	4.78
E.coli	1441	5871	8.14
H.pylori	687	1351	3.93
C.elegans	2387	3825	3.20

3.1 Robustness of the treelet distribution

In order to use the treelet distribution as a reliable means of comparing PPI network topologies, one needs to ensure that it is robust; i.e. it does not change much with respect to small alterations to the network. This is of key importance as the available PPI networks are neither complete nor error free. In fact, the PPI networks we use in this study are known to include only a subset of proteins in the respective organism and the interactions between them (Han *et al.*, 2005). Thus, we explore the robustness of the treelet distribution with respect to random sparsification as proposed in (Han *et al.*, 2005).

The sparsification process involves two parameters α_b and α_e which are defined as follows. α_b, *the bait sampling probability*, is the probability that a vertex is kept in the network during the sparsification process; α_e, *the edge sampling probability*, is the probability that an edge is kept in the network during sparsification.

We performed two independent experiments for evaluating the robustness of the treelet distribution on the Yeast PPI network, which is the best developed among the networks we considered. In the first experiment we set both α_b and α_e to 0.7, and in the second one we set both to 0.5. We performed *five* independent sparsifications of the Yeast network; their treelet distributions are provided in Figures 5 and 6.

In Hormozdiari *et al.* (2007) it was observed that sparsification as suggested by Han *et al.* (2005) has limited effect on the distribution of all subgraph topologies of up to *six* vertices, provided one focuses on induced occurrences of these subgraphs. Here, we obtain similar results on the robustness of the non-induced occurrences of trees of up to 10 vertices, for $\alpha_b = \alpha_e = 0.7$. However, for $\alpha_b = \alpha_e = 0.5$ significant variations can be observed.

Note that among the tree topologies we considered there is no 'natural' ordering that can be used in our distribution plots. As a result, we chose to use the ordering implied by the robustness of independent treelets with respect to the bait and edge sampling probability of 0.5. In all our plots, the 'robustness' of the treelets, i.e. the ratio between the minimum count and the maximum count of the treelets among the *five* sparsifications we obtained, decrease from left to right. In other words, a treelet with i.d. ℓ is at least as robust as (or more robust than) a treelet with i.d. $\ell+1$ for all values of ℓ.

3.2 Comparing PPI networks w.r.t. normalized treelet distribution

We first discuss how the normalized treelet distributions vary among the most complete PPI networks available via the *Database of Interacting Proteins* (Xenarios *et al.*, 2002). Figure 7 compares the normalized treelet distributions of the Yeast, *H.pylori* and *E.coli* PPI networks (in fact their largest connected components). Although all three PPI networks of unicellular organisms are quite similar with respect to normalized treelet distributions, it is interesting to note that the PPI network of the Yeast appears to be more similar to that of the *H.pylori* in comparison to the *E.coli* PPI network. We also compare the normalized treelet distributions of these three unicellular organisms' PPI network with that of the most complete PPI network of a multicellular organism, *C.elegans* in Figure 8. As can be seen, the normalized treelet distribution of *C.elegans* is very different from that of the Yeast, *E.coli* or *H.pylori*.

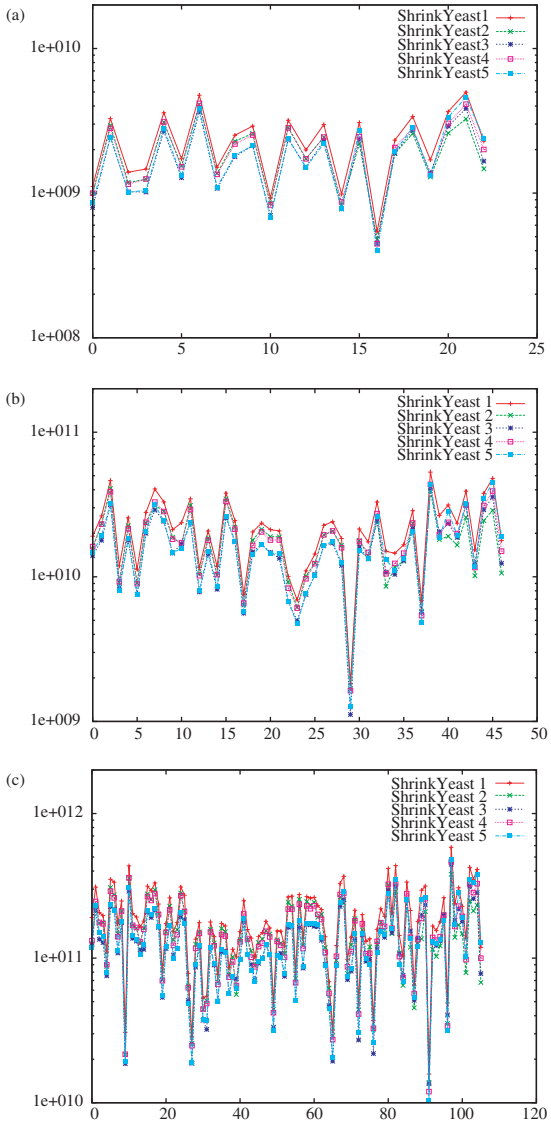

Fig. 5. A comparison of treelet distributions of five networks (a) size 8, (b) size 9 and (c) size 10 generated from the Yeast PPI network with both the bait and edge sampling probability equal to 0.7.

3.3 Comparing PPI networks with artificial networks w.r.t. normalized treelet distribution

We also compare the normalized treelet distribution of the PPI networks of the Yeast, *E.coli*, *H.pylori* with two artificial network

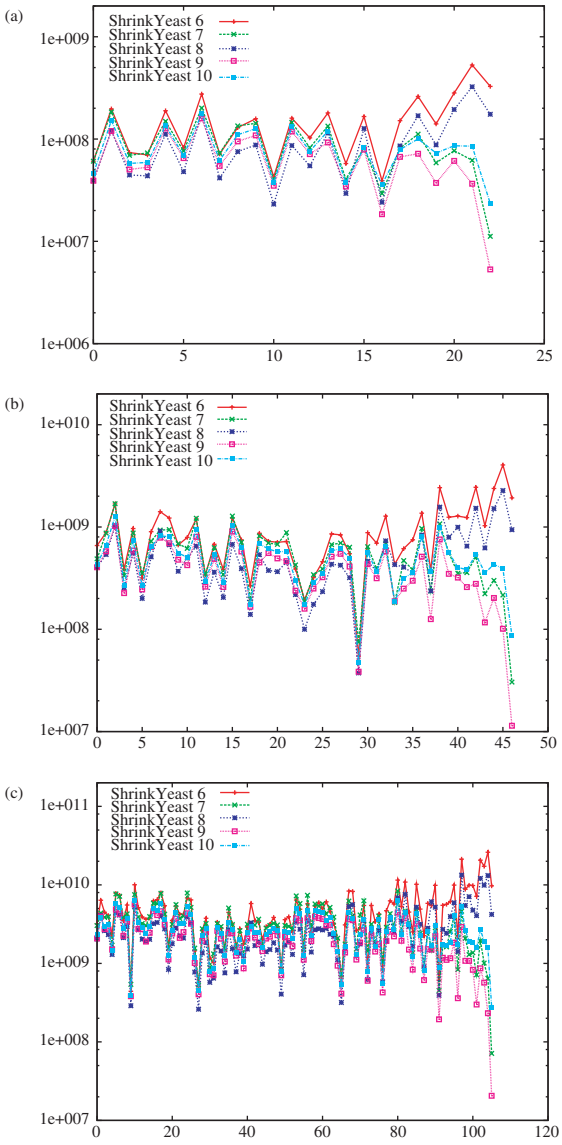

Fig. 6. A comparison of the treelet distributions of five networks (a) size 8, (b) size 9 and (c) size 10 generated from the Yeast PPI network with both the bait and edge sampling probability equal to 0.5.

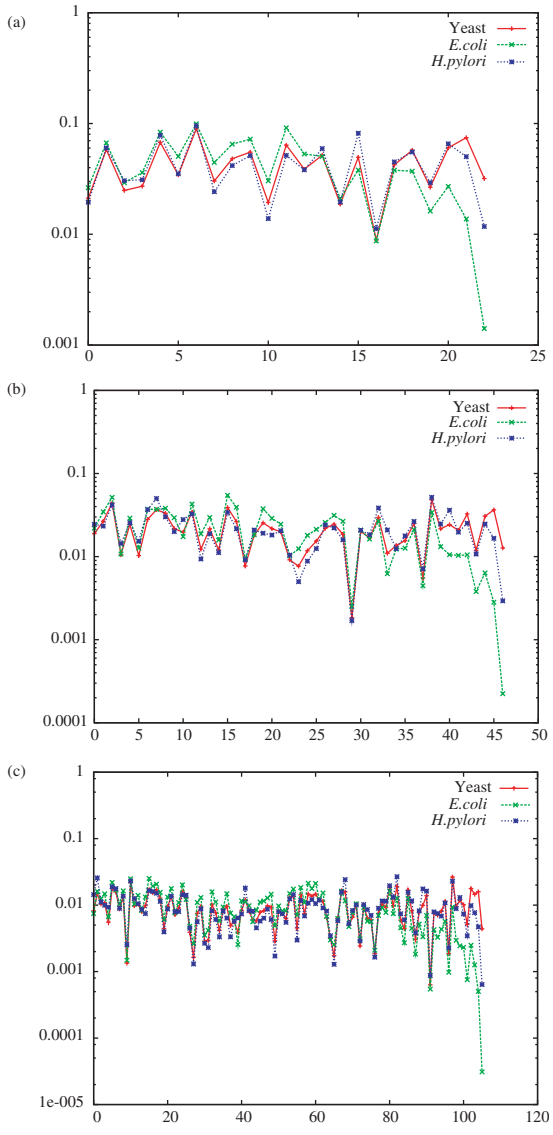

Fig. 7. Normalized treelet distribution (a) size 8, (b) size 9 and (c) size 10 of the Yeast PPI network (Red), *E.coli* (Green) and *H.pylori* (Blue).

models that have been proposed to emulate the evolution of PPI networks.[4] Such a comparison not provides insight into the evolution of PPI networks but also may help detect network motifs. Among these models, the preferential attachment model starts with a small seed network and grows by adding to the network one vertex in each iteration. The new vertex is connected to every other vertex independently with a fixed probability that determines the average degree of the vertices. As per the available PPI networks, the preferential attachment model is known to generate networks with power-law degree distribution.

[4]Note that the well-known Erdos–Renyi random graph model (Erdos and Renyi, 1959) is very different from the ones we consider here with respect to the graphlet distribution, as shown in Hormozdiari *et al.* (2007) for $k < 7$.

The duplication model again starts with a small seed network and grows by 'duplicating' a randomly picked vertex with all its links (Bebek *et al.*, 2006; Chung *et al.*, 2003; Hormozdiari *et al.*, 2007). Then each link is deleted with some fixed probability. This is followed by establishing links with every other vertex, again with constant probability. As per the above networks, duplication model generates power-law degree distributions.

In Hormozdiari *et al.* (2007) it was observed that the duplication model better emulates the available PPI networks with respect to k-hop reachability, closeness, betweenness and graphlet distribution with up to 6 vertices, in comparison to the preferential attachment model. Here, again we consider the specific duplication and preferential attachment models, whose parameters are chosen so as to approximate the degree distribution of the Yeast network as much as possible.

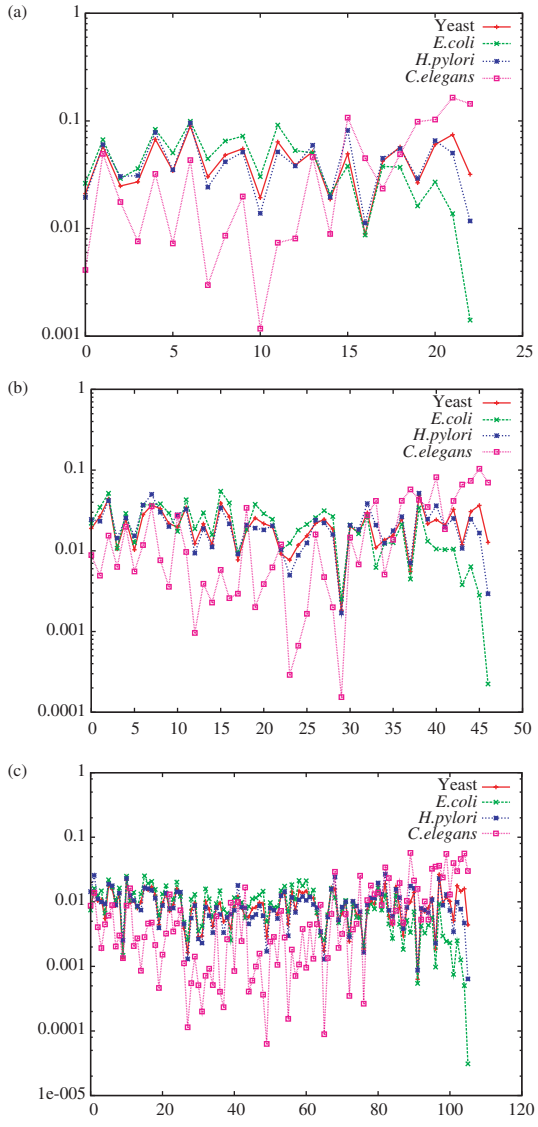

Fig. 8. Normalized treelet (a) size 8, (b) size 9 and (c) size 10 distribution of the Yeast PPI network (Red), *E.coli* (Green), *H.pylori* (Blue) and *C.elegans* (Pink).

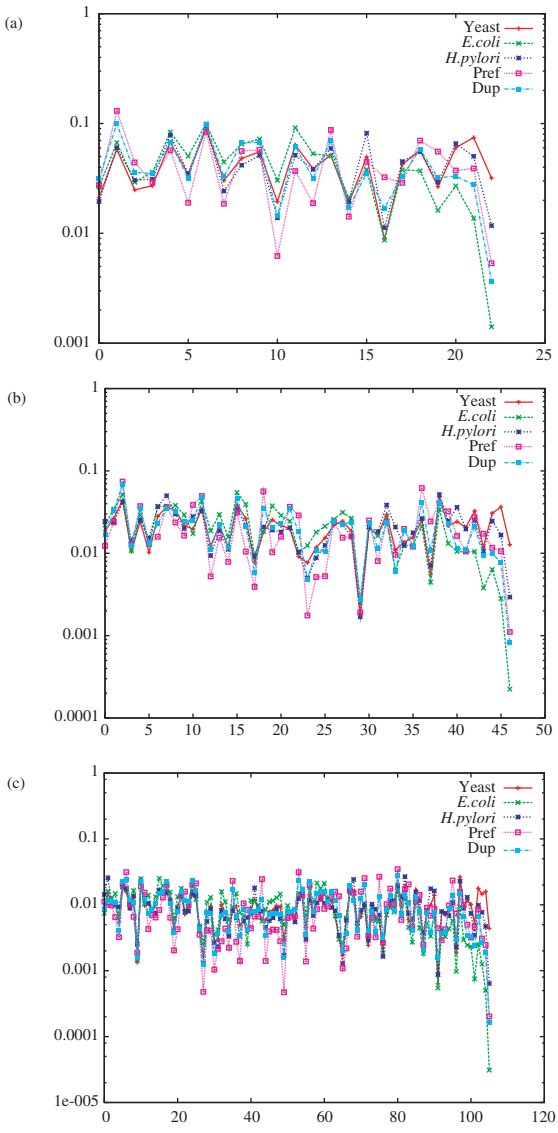

Fig. 9. Normalized treelet distribution (a) size 8, (b) size 9 and (c) size 10 of the Yeast (Red), *H.pylori* (Blue), *E.coli* (Green) PPI networks and Preferential Attachment model (Pink), Duplication model (Cyan).

The single parameter available for the preferential attachment model determines the average degree of the network. As our goal is to emulate the Yeast PPI network as much as possible, we fix this parameter so as to make the average degree in the Yeast PPI network and the artificial network equal. The duplication model has two parameters which determine not only the average degree of the network generated but also its degree distribution. We fix these two parameters so that the average degree and the degree distribution of the artificial network is 'as similar to' that of the Yeast PPI network as possible.

After generating artificial networks via the above network models, we applied our algorithm to count the number of non-induced occurrences of all possible treelets in the Yeast, *E.coli* and *H.pylori* as well as the the artificial networks (in fact the average normalized distribution of five independent networks generated by these two models). The results are given in Figure 9.

It can be observed that the treelet distributions of all five networks are not far from each other. However, the one generated by the duplication model is much closer to the PPI networks; in all three plots, the distribution of the preferential attachment model can be seen to vary the most with respect to the other networks.

4 CONCLUSION

Recently developed algorithms such as Przulj *et al.* (2004) Grochow and Kellis (2007), Hormozdiari *et al.* (2007), have been able to count the number of *induced* occurrences of subgraphs with ≤ 7 vertices in available PPI networks. Unfortunately as the number of vertices increase, the running time of these algorithms grow exponentially

and thus they become impractical. Furthermore, such algorithms can not count *non-induced* occurrences of subgraphs. It is indeed possible that many non-induced occurrences of a subgraph G' share exactly the same vertex set in a graph G. Thus, the number of non-induced occurrences of a subgraph is many times more than the number of induced occurrences. Counting non-induced occurrences of sizable network motifs is not only challenging but also quite desirable as available PPI networks include many false and missing interactions.

This article addresses both of the above challenges provided that the subgraphs in question are in the form of trees or bounded treewidth graphs. We show how to apply color coding technique to count the number of non-induced occurrences of such subgraphs in time polynomial with n if $k = O(\log n)$.

We used our algorithm to obtain 'treelet' distributions for $k \leq 10$ of the PPI networks of unicellular organisms (*S.cerevisiae*, *E.coli* and *H.pylori*), which are all quite similar, and a multicellular organism (*C.elegans*) which is significantly different. Furthermore the treelet distribution of the unicellular organisms are similar to that obtained by the 'duplication model' but are quite different from that of the 'preferential attachment model'. The treelet distribution is robust w.r.t. random sparsification with bait/edge coverage of 70% but differences can be observed when bait/edge coverage drops to 50%.

Conflict of Interest: none declared.

REFERENCES

Alon,N. and Gutner,S. (2007) Balanced families of perfect hash functions and their applications. *Proc. ICALP*, pp. 435–446.

Alon,N. *et al.* (1995) Color-coding. *J. ACM*, **42**, 844–856.

Arvind,V. and Raman,V. (2002) Approximation algorithms for some parameterized counting problems. In *Proceedings of the 13th International Symposium on Algorithms and Computation* (ISAAC '02). pp. 453–464.

Barabási,A.L. and Albert,R. (1999) Emergence of scaling in random networks. *Science*, **286**, 509–512.

Bebek,G. *et al.* (2006) The degree distribution of the generalized duplication model. *Theor. Comput. Sci.*, **369**, 239–249.

Bollobás,B. *et al.* (2001) The degree sequence of a scale-free random graph process. *Random Struct. Algorithms*, **18**, 279–290.

Chung,F. *et al.* (2001) A random graph model for power law graphs. *Experimental Math.*, **10**, 53–66.

Chung,F. *et al.* (2003) Duplication models for biological networks. *J. Comput. Biol.*, **10**, 677–687.

Dost,B. *et al.* (2007) Qnet: a tool for querying protein interaction networks. In *RECOMB*, pp. 1–15.

Eisenberg,E. and Levanon,E.Y. (2003) Preferential attachment in the protein network evolution. *Phys. Rev. Lett.*, **91**.

Erdos,P. and Renyi,A. (1959) On random graphs. *Publicationes Mathematicae*, **6**, 290–297.

Grochow,J.A. and Kellis,M. (2007) Network motif discovery using subgraph enumeration and symmetry-breaking. In *RECOMB*, pp. 92–106.

Han,J. *et al.* (2005) Effect of sampling on topology predictions of protein–protein interaction networks. *Nat. Biotech.*, **23**, 839–844.

Hormozdiari,F. *et al.* (2007) Not all scale-free networks are born equal: the role of the seed graph in ppi network evolution. *PLoS Comput. Biol.*, **3**.

Karp,R.M. and Luby,M. (1983) Monte-carlo algorithms for enumeration and reliability problems. In *FOCS*, pp. 56–64.

Kashtan,N. *et al.* (2004) Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs. *Bioinformatics*, **20**, 1746–1758.

Milo,R. *et al.* (2002) Network motifs: simple building blocks of complex networks. *Science*, **298**, 824–827.

Przulj,N. *et al.* (2004) Modeling interactome: scale-free or geometric? *Bioinformatics*, **20**, 3508–3515.

Scott,J. *et al.* (2006) Efficient algorithms for detecting signaling pathways in protein interaction networks. *J. Comput. Biol.*, **13**, 133–144.

Shlomi,T. *et al.* (2006) Qpath: a method for querying pathways in a protein–protein interaction network. *BMC Bioinformatics*, **7**, 199.

Vázquez,A. *et al.* (2003) Modelling of protein interaction networks. *Complexus*, **1**, 38–44.

Xenarios,I. *et al.* (2002) Dip, the database of interacting proteins: a research tool for studying cellular networks of protein interactions. *Nucl. Acids Res.*, **30**, 303–305.